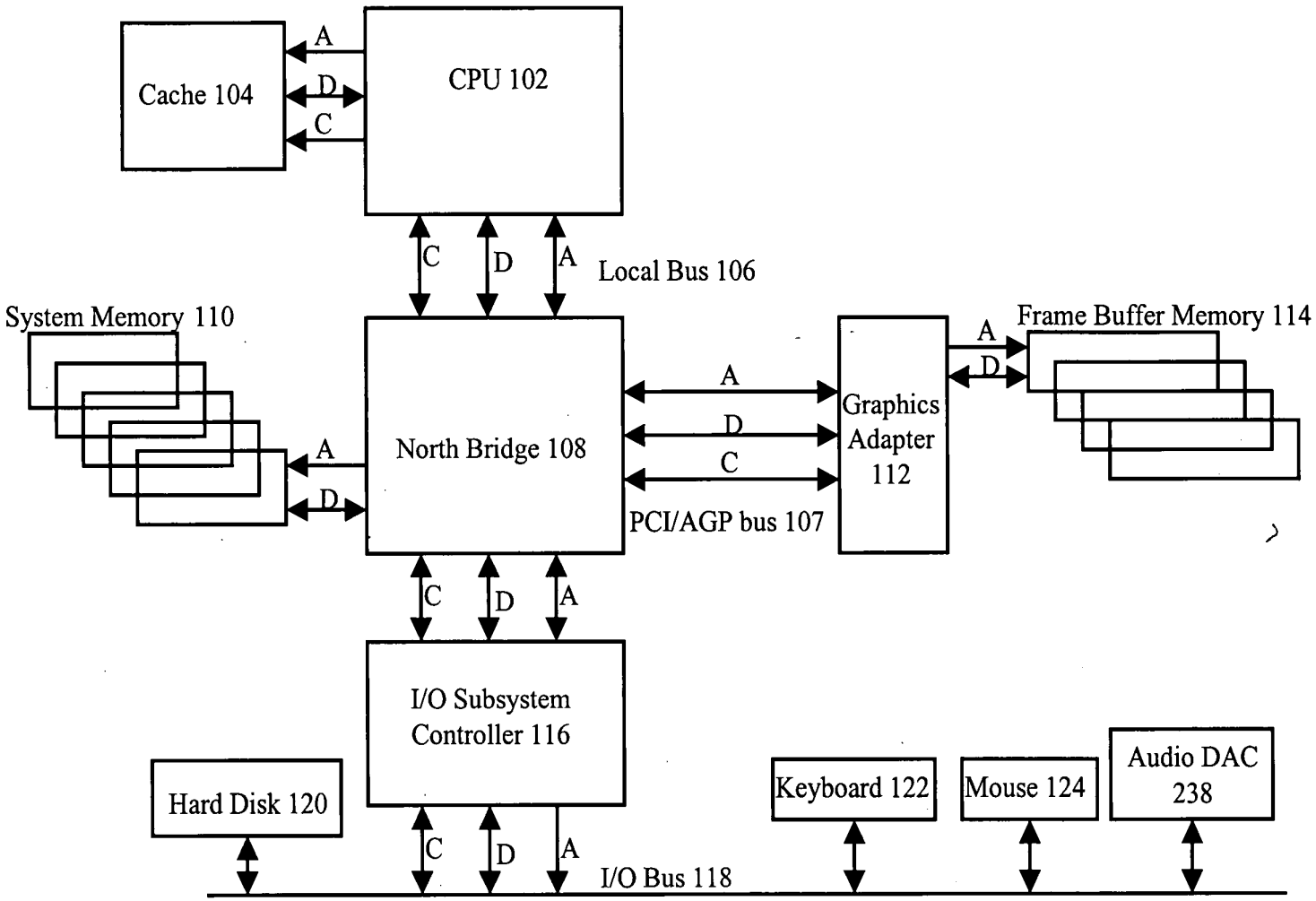


Figure 1
Prior Art

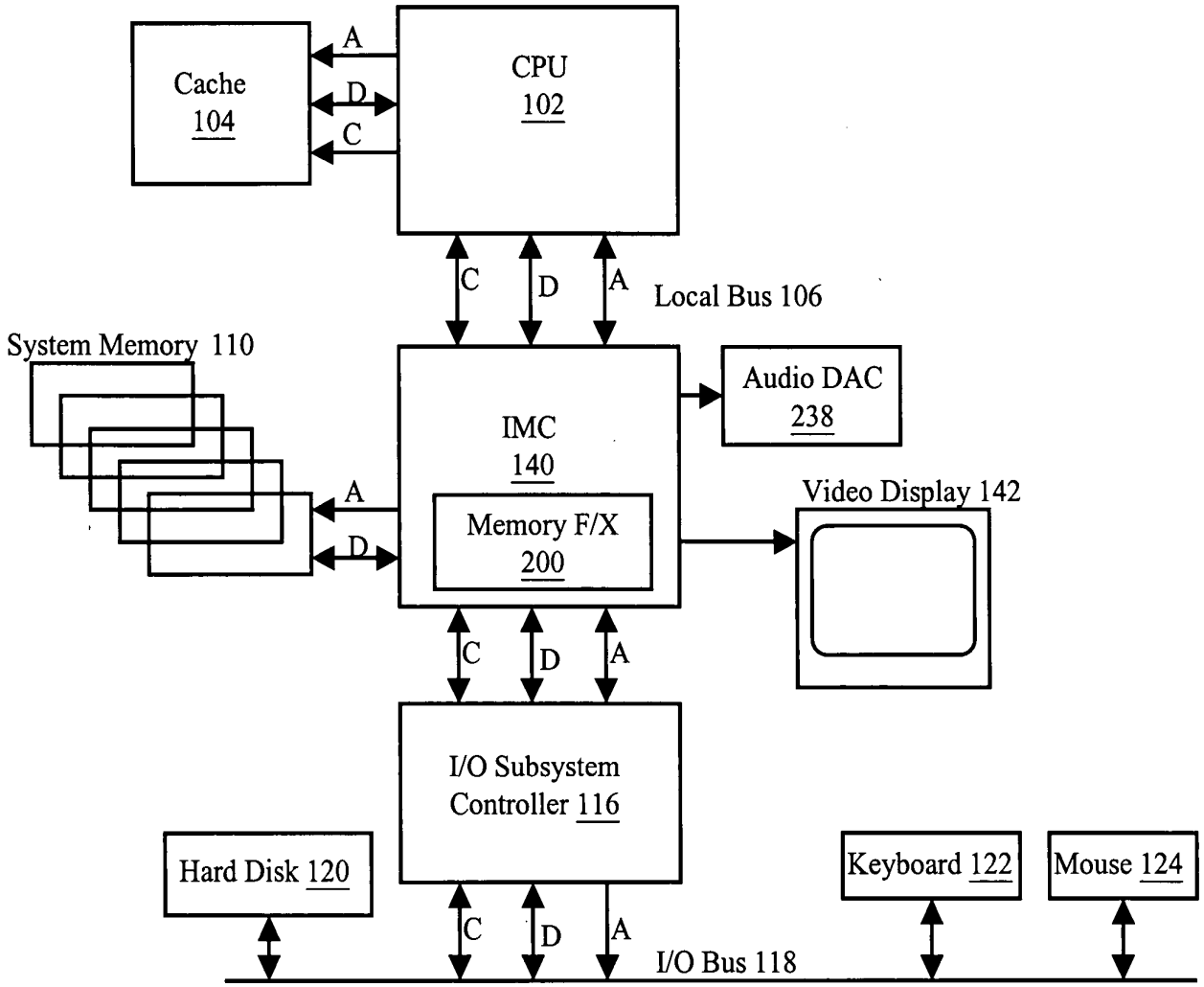

Cache
104
A
D
C
CPU
102
C
D
A
Local Bus 106
System Memory 110
IMC
140
Audio DAC
238
A
D
Memory F/X
200
Video Display 142
C
D
A
I/O Subsystem
Controller 116
Hard Disk 120
Keyboard 122
Mouse 124
C
D
A
I/O Bus 118

Figure 2A

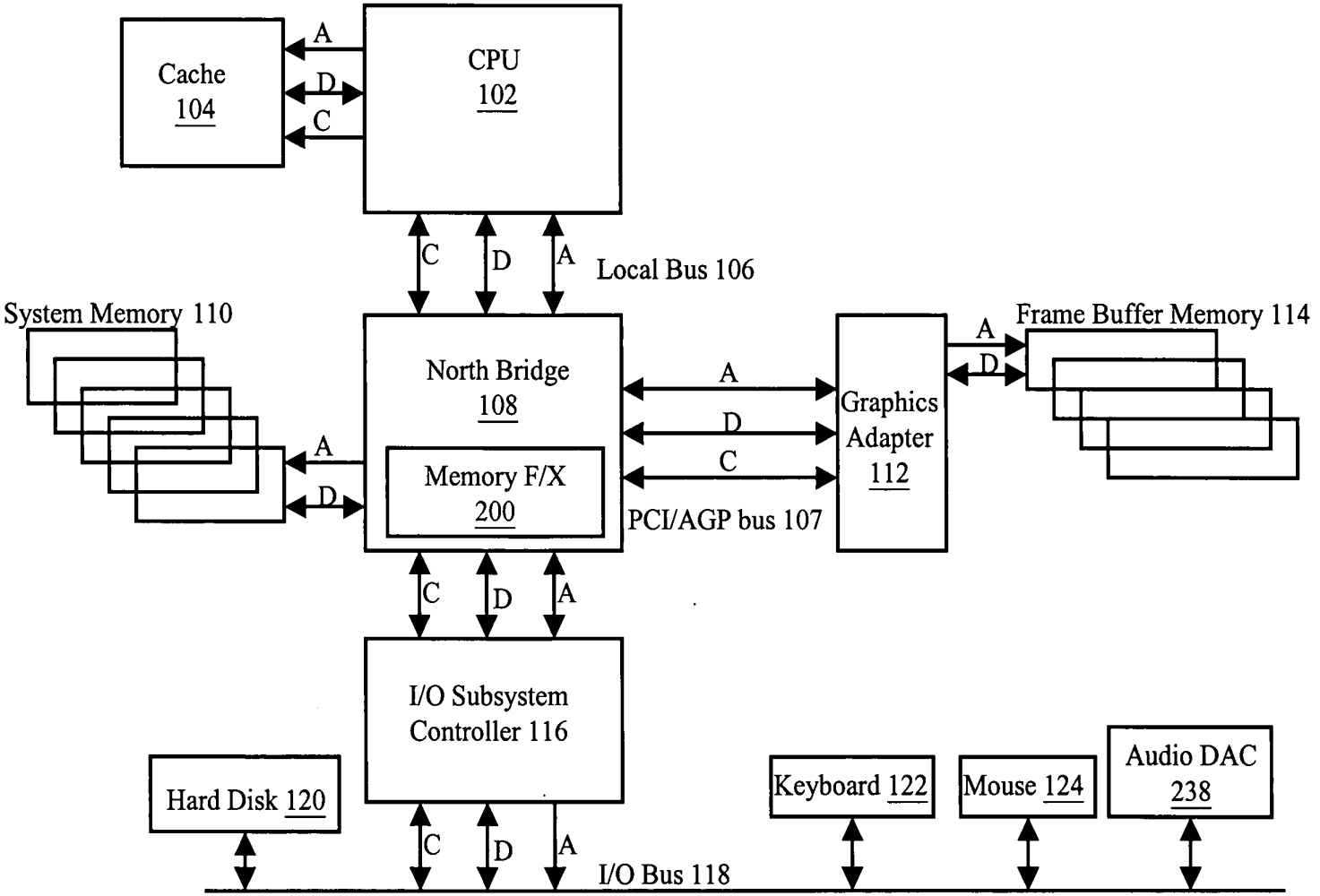

Cache
104
A
D
C
CPU
102
C
D
A
Local Bus 106
System Memory 110
North Bridge
108
Memory F/X
200
A
D
A
D
C
PCI/AGP bus 107
Graphics
Adapter
112
A
D
Frame Buffer Memory 114
C
D
A
I/O Subsystem
Controller 116
Hard Disk 120
C
D
A
I/O Bus 118
Keyboard 122
Mouse 124
Audio DAC
238

Figure 2B

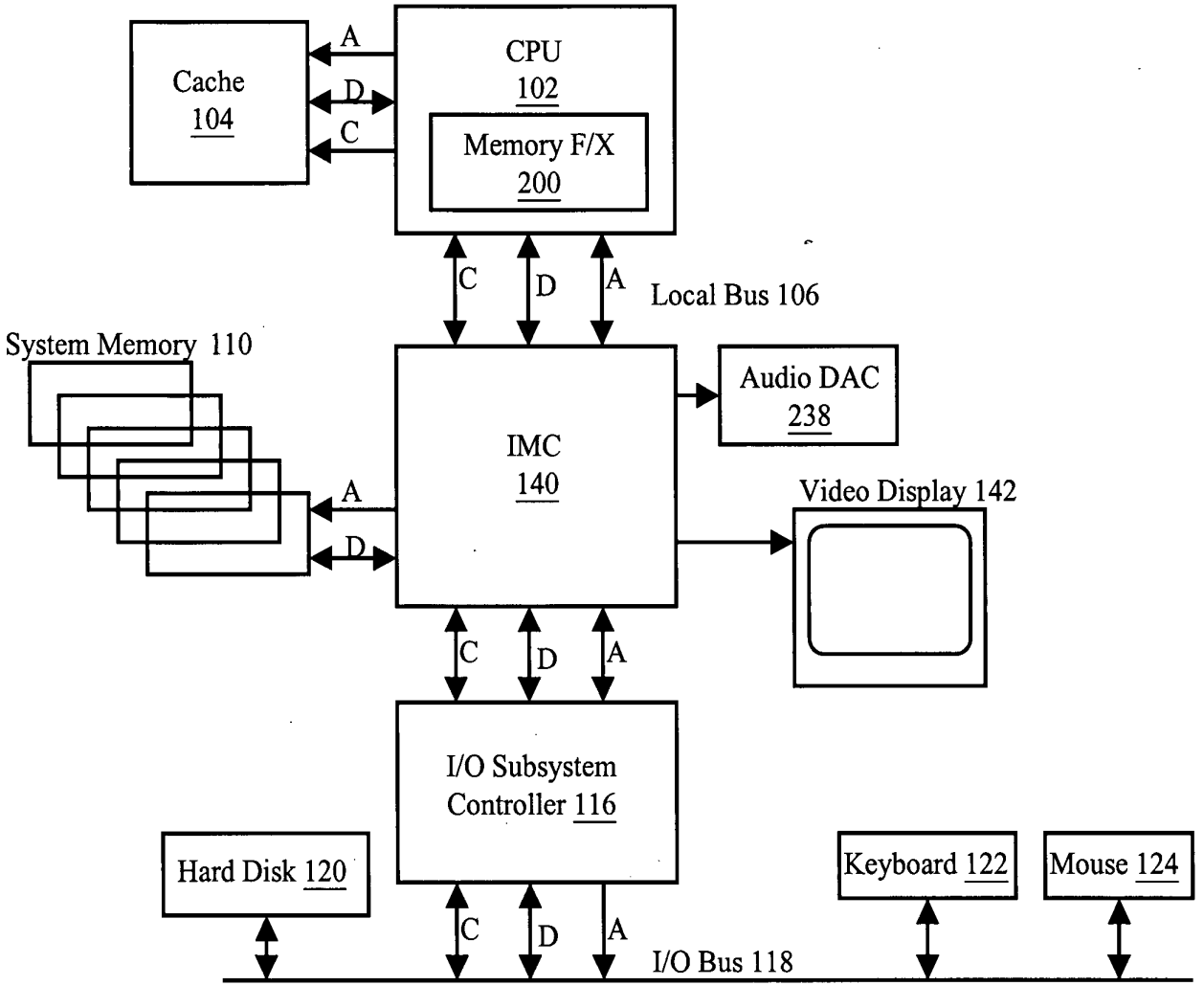

Cache
104
A
D
C
CPU
102
Memory F/X
200
C
D
A
Local Bus 106
System Memory 110
IMC
140
Audio DAC
238
A
D
Video Display 142
C
D
A
I/O Subsystem
Controller 116
Hard Disk 120
Keyboard 122
Mouse 124
C
D
A
I/O Bus 118

Figure 2C

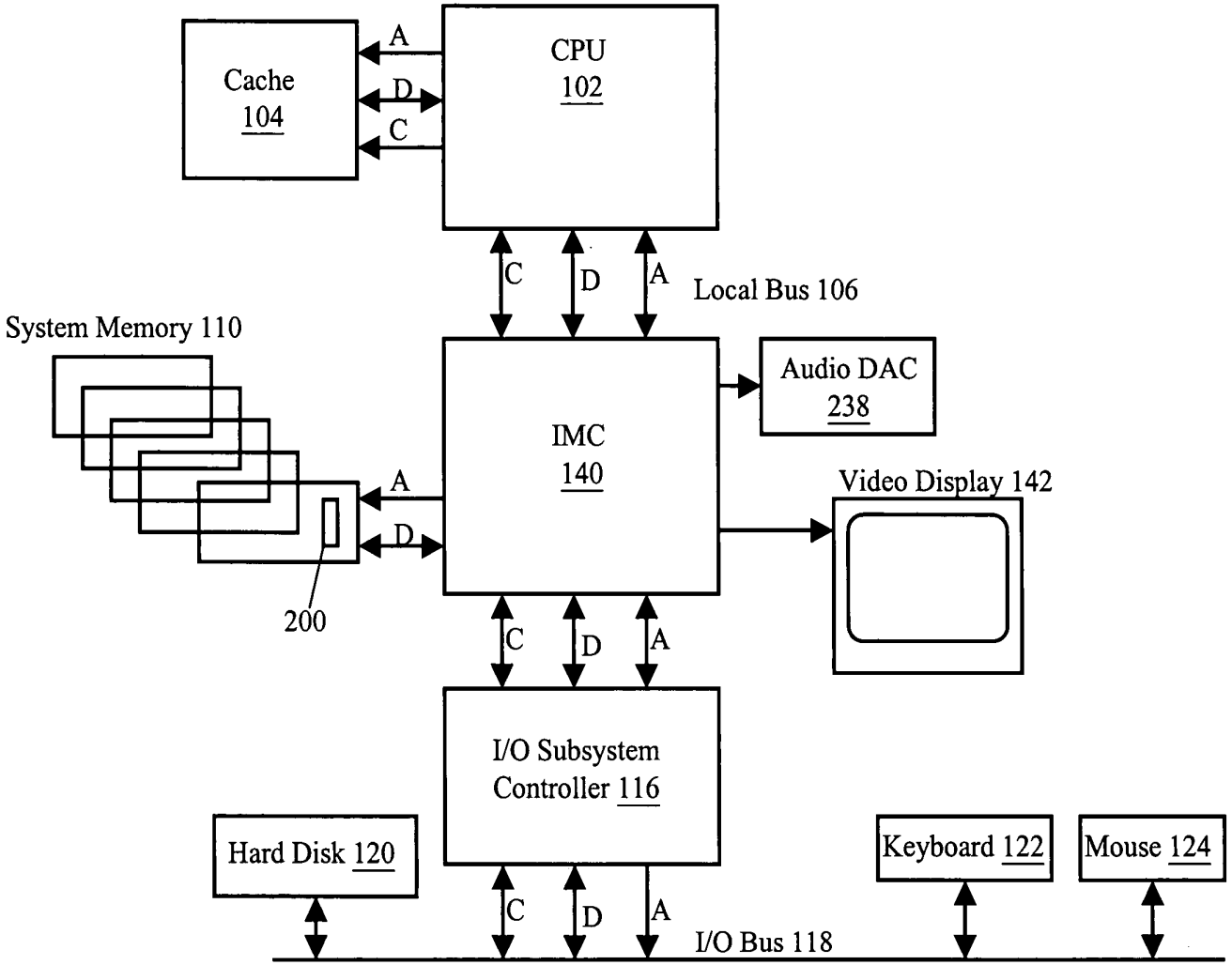

Cache
104
A
D
C
CPU
102
C
D
A
Local Bus 106
System Memory 110
Audio DAC
238
IMC
140
A
D
Video Display 142
200
C
D
A
I/O Subsystem
Controller 116
Hard Disk 120
Keyboard 122
Mouse 124
C
D
A
I/O Bus 118

Figure 2D

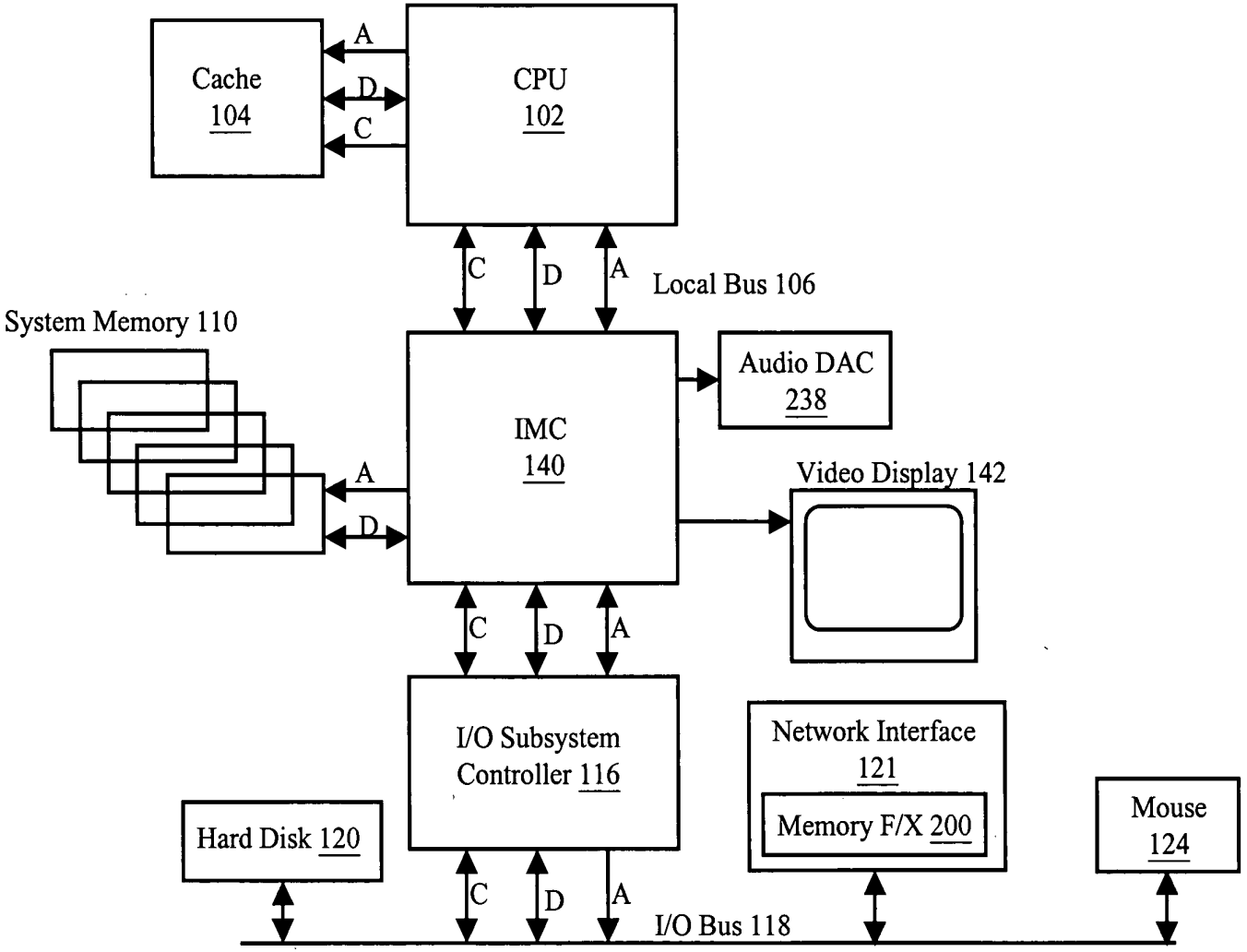

Cache
104
A
D
C
CPU
102
C
D
A
Local Bus 106
System Memory 110
IMC
140
Audio DAC
238
A
D
Video Display 142
C
D
A
I/O Subsystem
Controller 116
Network Interface
121
Memory F/X 200
Mouse
124
Hard Disk 120
C
D
A
I/O Bus 118

Figure 2E

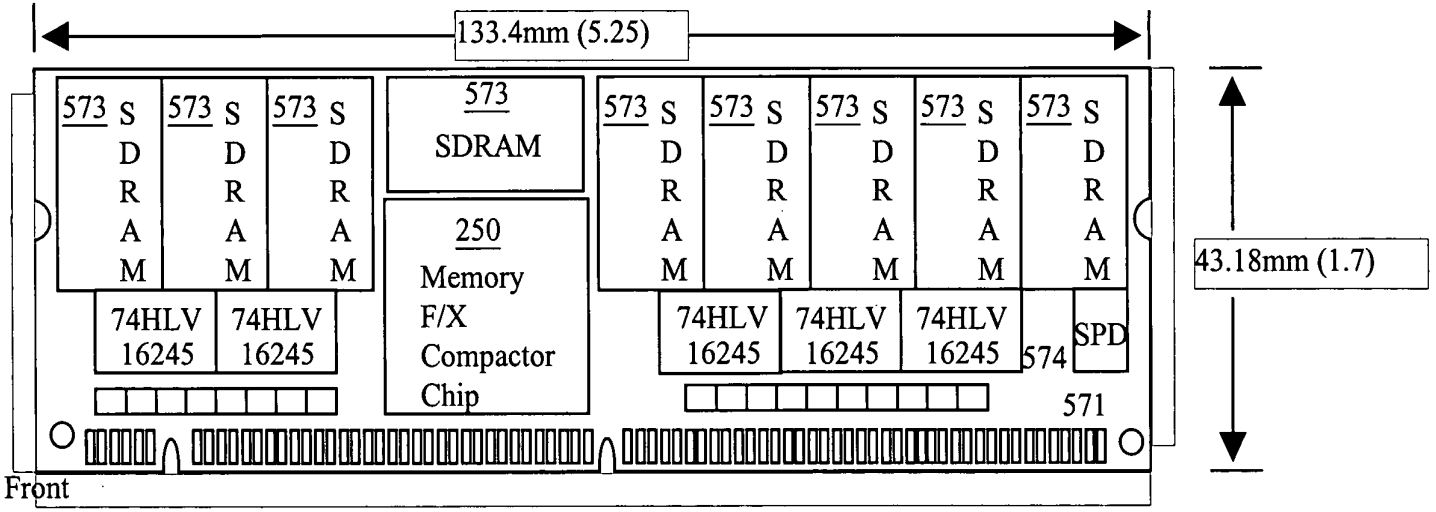


Figure 3A

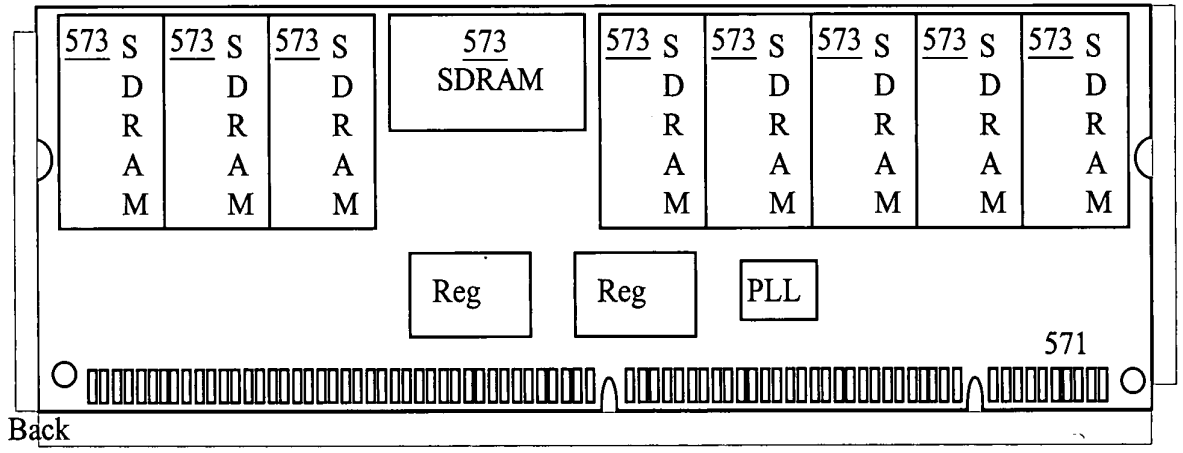
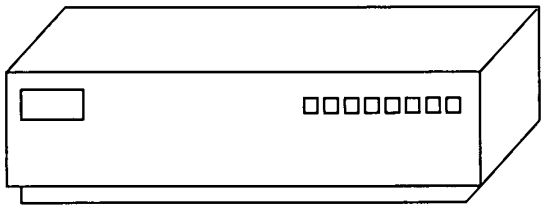
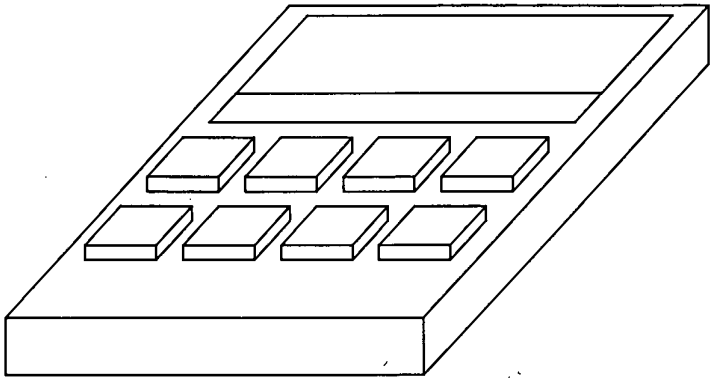


Figure 3B



Router 130

Figure 4



PDA

Figure 5

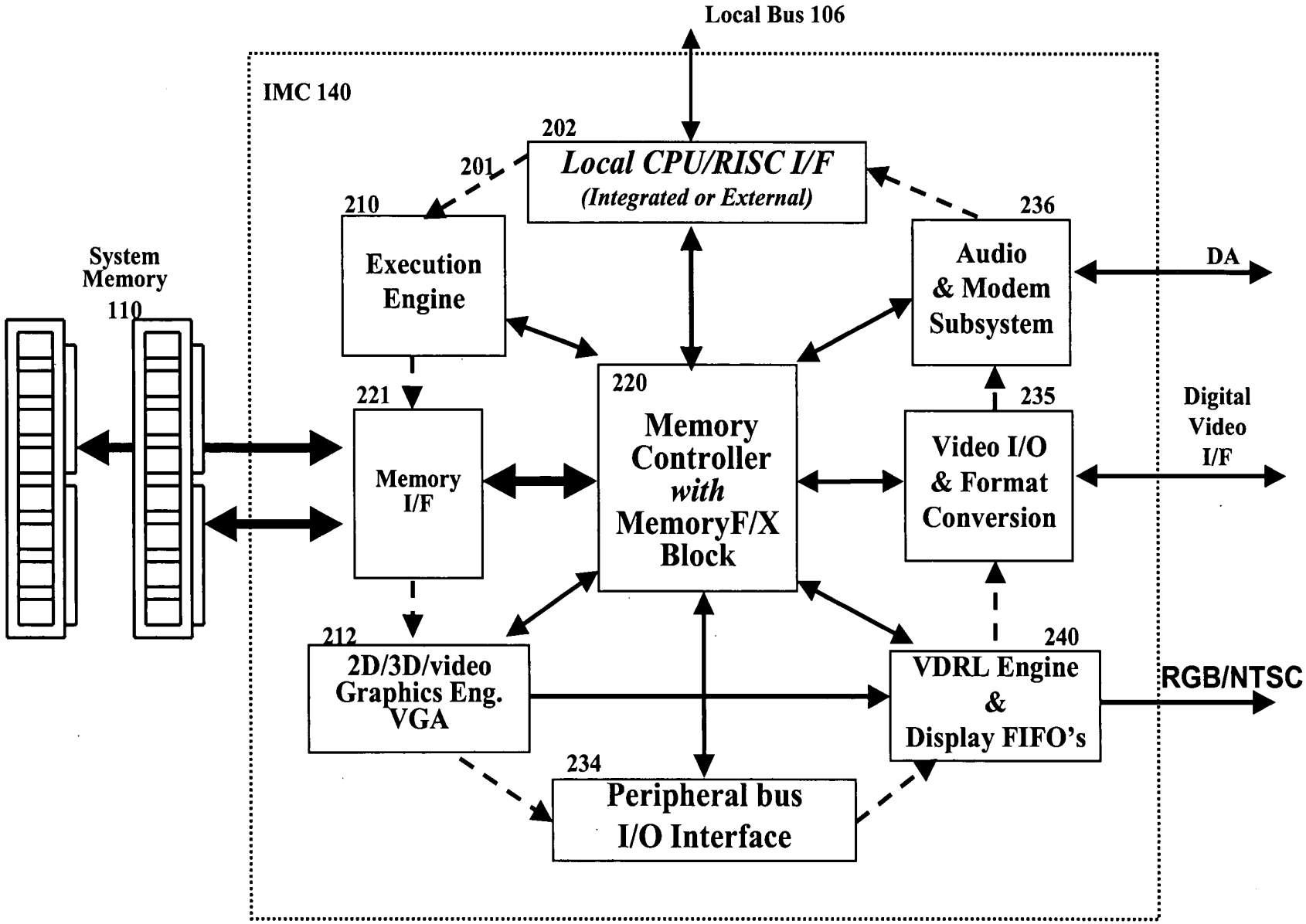

Local Bus 106
IMC 140
202
201
Local CPU/RISC I/F
(Integrated or External)
210
Execution Engine
236
Audio & Modem Subsystem
DA
System Memory
110
221
220
Memory Controller with MemoryF/X Block
235
Digital Video I/F
Memory I/F
Video I/O & Format Conversion
212
240
2D/3D/video Graphics Eng. VGA
VDRL Engine & Display FIFO's
RGB/NTSC
234
Peripheral bus I/O Interface

Figure 6

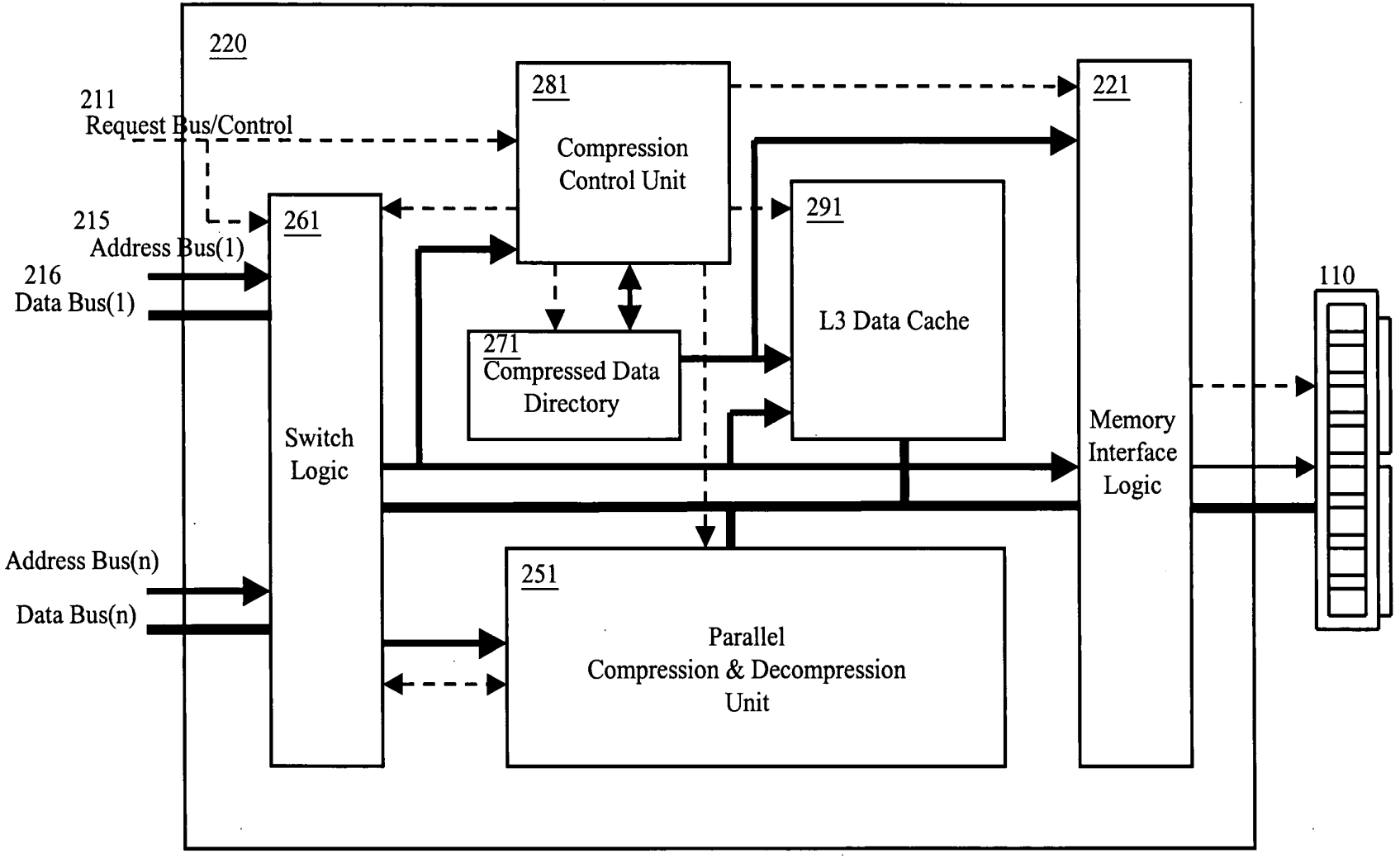
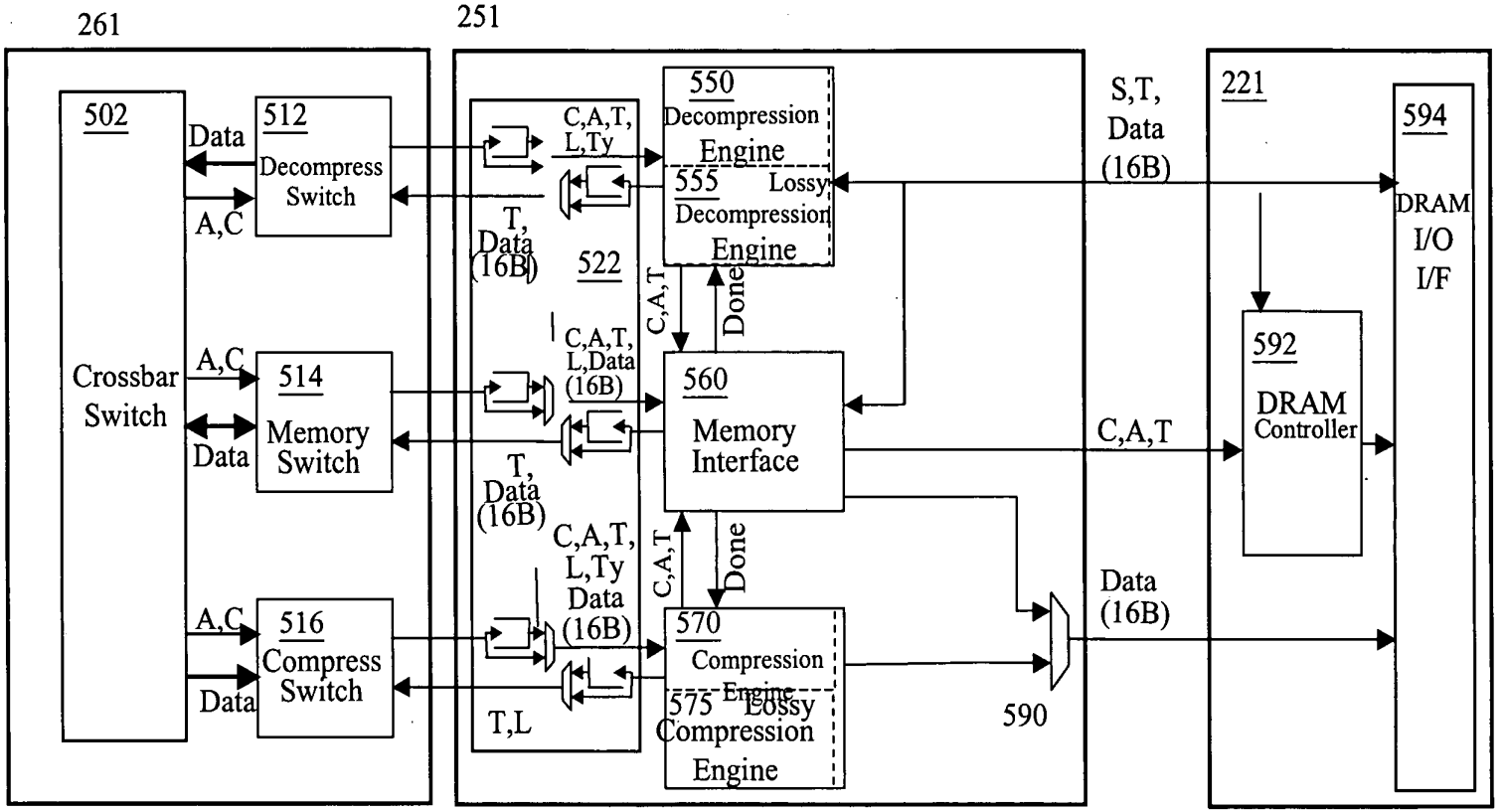

220
211
Request Bus/Control
281
Compression
Control Unit
221
215
Address Bus(1)
261
291
216
Data Bus(1)
L3 Data Cache
110
271
Compressed Data
Directory
Switch
Logic
Memory
Interface
Logic
Address Bus(n)
Data Bus(n)
251
Parallel
Compression & Decompression
Unit

Figure 7



Key:
C – Command
A – Address
T – Tag
L – Length
Ty – Type
S – Status

Figure 8

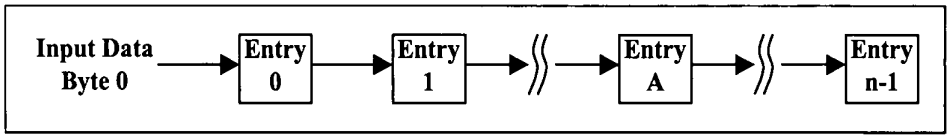


Figure 9A, Prior Art

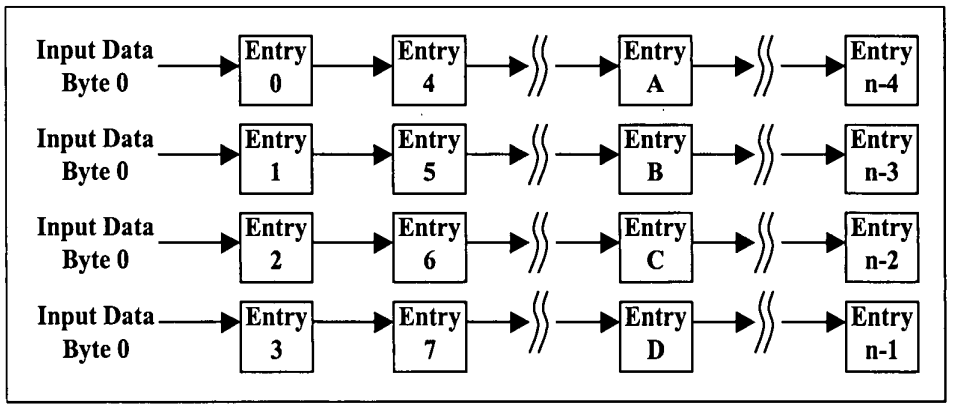


Figure 9B, New Art

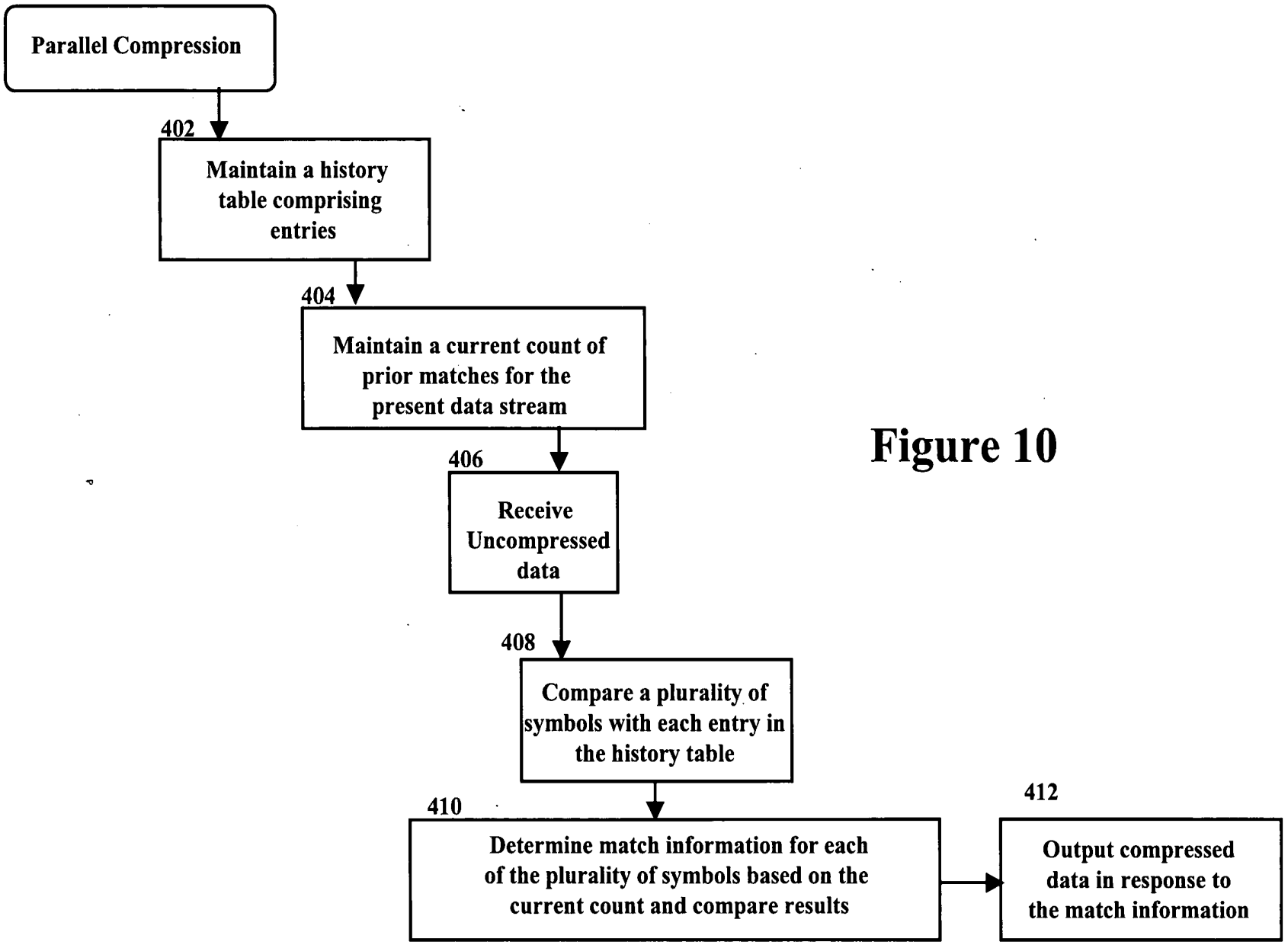

Parallel Compression
402
Maintain a history table comprising entries
404
Maintain a current count of prior matches for the present data stream
406
Receive Uncompressed data
408
Compare a plurality of symbols with each entry in the history table
410
Determine match information for each of the plurality of symbols based on the current count and compare results
412
Output compressed data in response to the match information

Figure 10

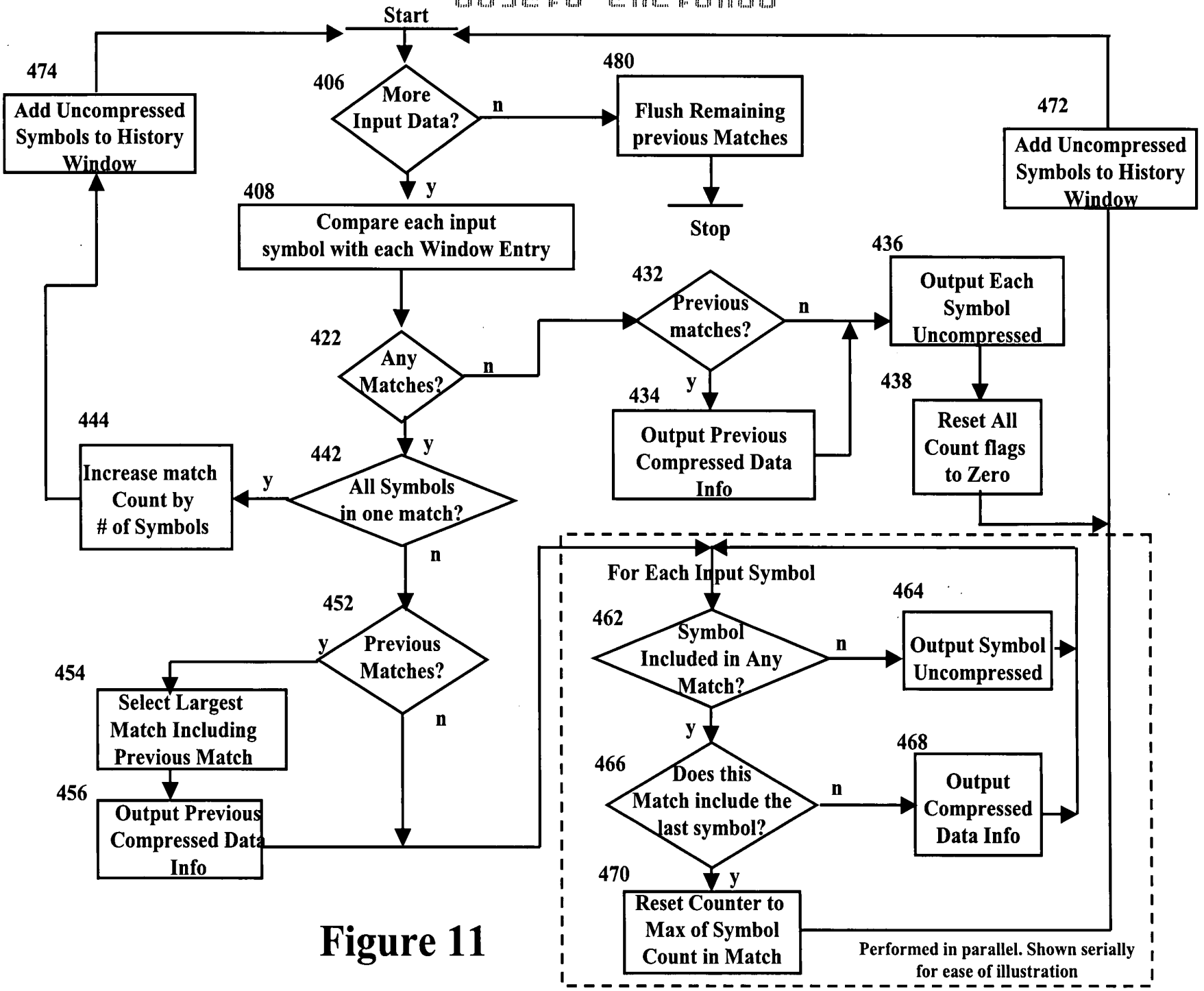

Start
474
Add Uncompressed Symbols to History Window
406
More Input Data?
n
480
Flush Remaining previous Matches
Stop
472
Add Uncompressed Symbols to History Window
y
408
Compare each input symbol with each Window Entry
422
Any Matches?
n
432
Previous matches?
n
436
Output Each Symbol Uncompressed
y
434
Output Previous Compressed Data Info
438
Reset All Count flags to Zero
444
Increase match Count by # of Symbols
y
442
All Symbols in one match?
y
n
452
Previous Matches?
y
n
454
Select Largest Match Including Previous Match
456
Output Previous Compressed Data Info
For Each Input Symbol
462
Symbol Included in Any Match?
n
464
Output Symbol Uncompressed
y
466
Does this Match include the last symbol?
n
468
Output Compressed Data Info
470
y
Reset Counter to Max of Symbol Count in Match
Performed in parallel. Shown serially for ease of illustration

Figure 11

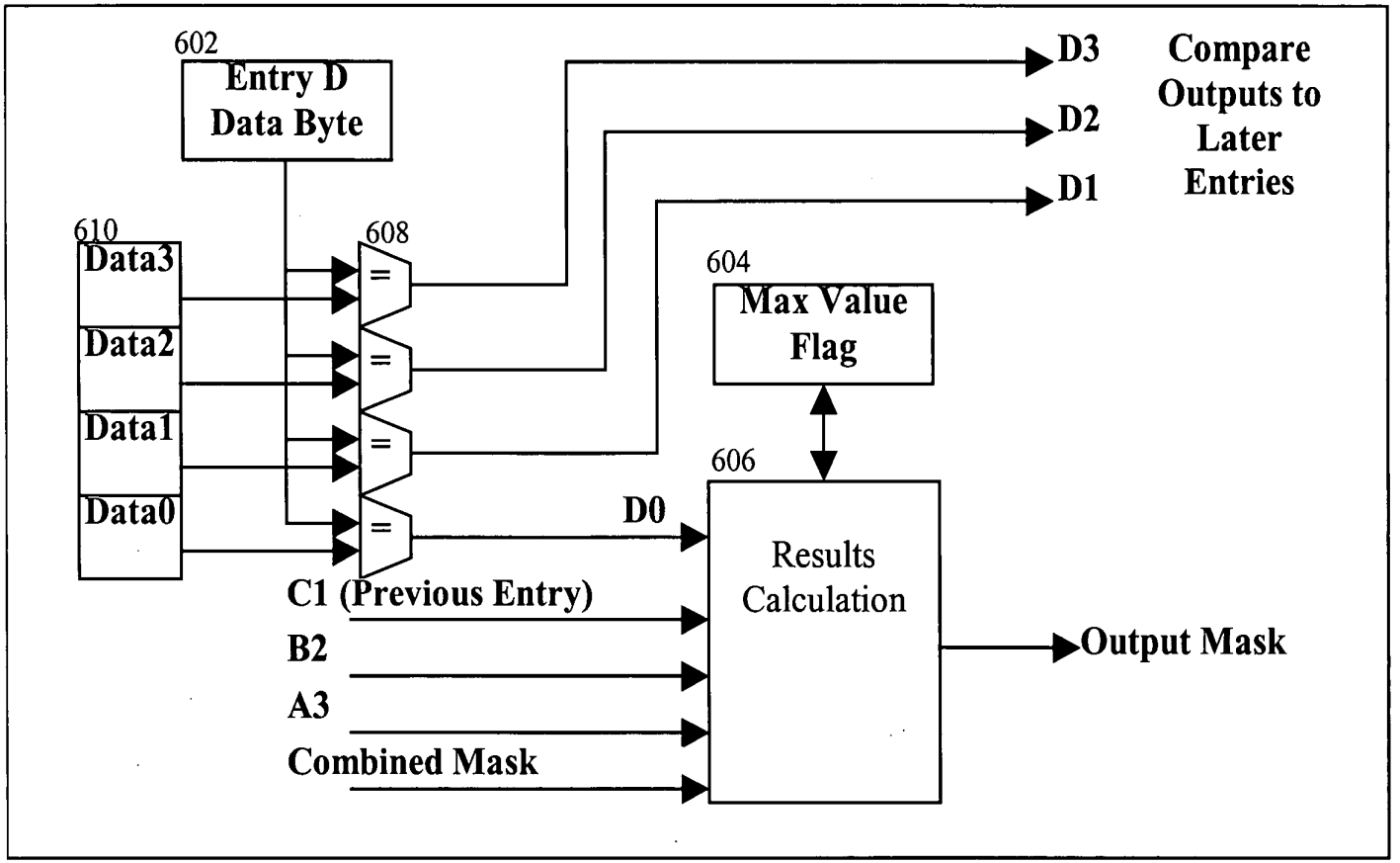

602
Entry D
Data Byte
610
Data3
Data2
Data1
Data0
608
=
=
=
=
D3
D2
D1
Compare
Outputs to
Later
Entries
604
Max Value
Flag
606
D0
Results
Calculation
Output Mask
C1 (Previous Entry)
B2
A3
Combined Mask

Figure 12

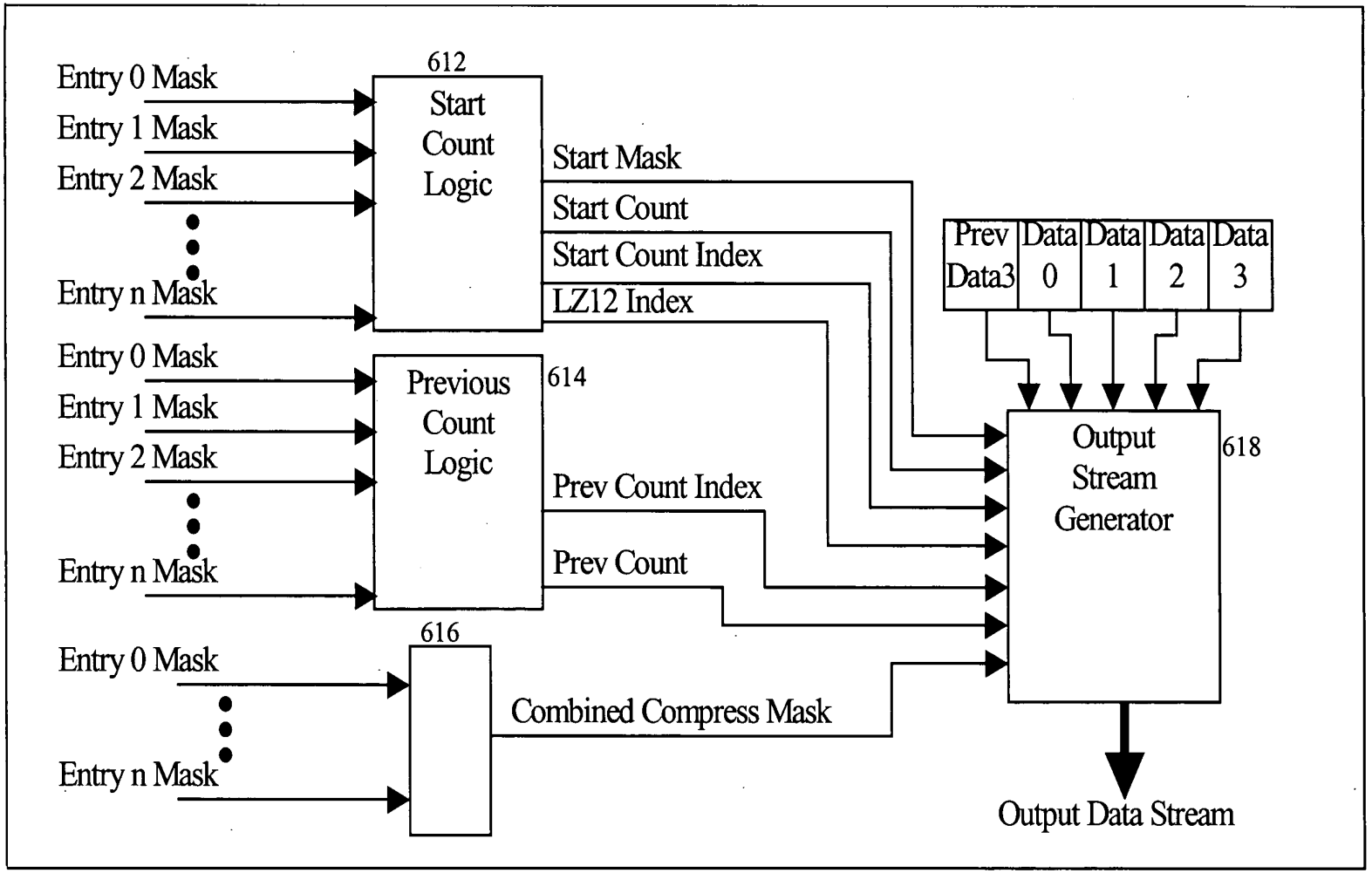

Entry 0 Mask
Entry 1 Mask
Entry 2 Mask
Entry n Mask
612
Start Count Logic
Start Mask
Start Count
Start Count Index
LZ12 Index
Entry 0 Mask
Entry 1 Mask
Entry 2 Mask
Entry n Mask
Previous Count Logic
614
Prev Count Index
Prev Count
Entry 0 Mask
Entry n Mask
616
Combined Compress Mask
Prev Data3
Data 0
Data 1
Data 2
Data 3
Output Stream Generator
618
Output Data Stream

Figure 13

Input Matches				Output
D0	C1	B2	A3	Mask
1	1	1	1	1111
1	1	1	0	1110
1	1	0	1	1101
1	1	0	0	1100
1	0	1	1	1011
1	0	1	0	1010
1	0	0	1	1001
1	0	0	0	1000
0	1	1	1	0111
0	1	1	0	0110
0	1	0	1	0101
0	1	0	0	0100
0	0	1	1	0011
0	0	1	0	0010
0	0	0	1	0001
0	0	0	0	0000

Figure 14a

Combined Mask	Count
0000	0
0001	1
0010	0
0011	2
0100	0
0101	1
0110	0
0111	3
1000	0
1001	1
1010	0
1011	2
1100	0
1101	1
1110	0
1111	Count+4

Figure 14b

Output Masks	Combined Mask
M1234 | 4321 &~M1	0001
432 & ~M12	0010
43 & ~M123	0100
4 & ~M1234	1000
First valid row determines Combined Mask Output	
M-Max Count Flag 1-1st Symbol Match 2-2nd Symbol Match 3-3rd Symbol Match 4-4th Symbol Match	

Figure 14c

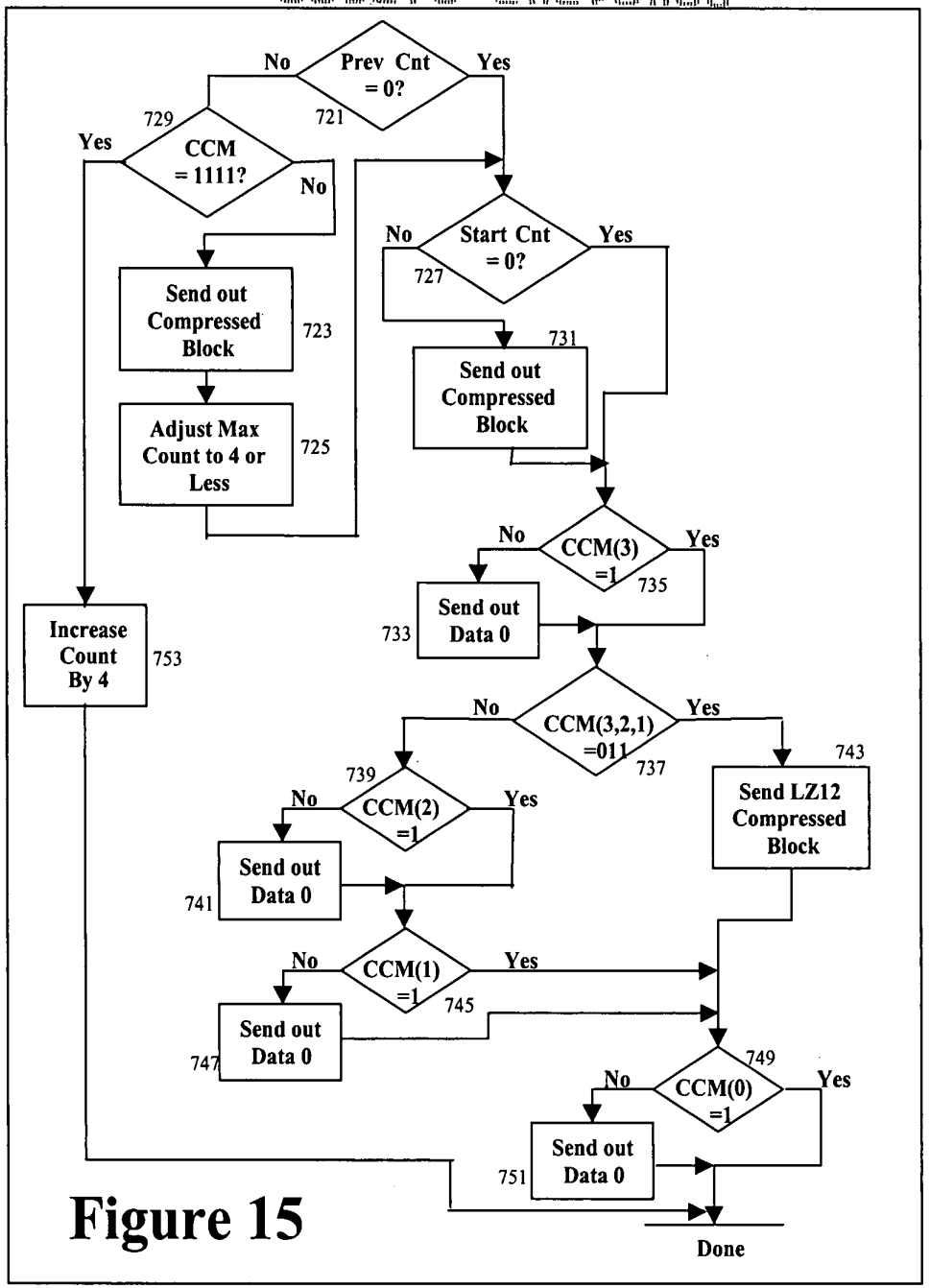

Prev Cnt = 0?
No
Yes
721
729
CCM = 1111?
Yes
No
Send out Compressed Block
723
Adjust Max Count to 4 or Less
725
Start Cnt = 0?
No
Yes
727
731
Send out Compressed Block
CCM(3) =1
No
Yes
735
Send out Data 0
733
Increase Count By 4
753
CCM(3,2,1) =011
No
Yes
737
743
Send LZ12 Compressed Block
739
CCM(2) =1
No
Yes
Send out Data 0
741
CCM(1) =1
No
Yes
745
Send out Data 0
747
749
CCM(0) =1
No
Yes
Send out Data 0
751
Done

Figure 15

09270 ETE660

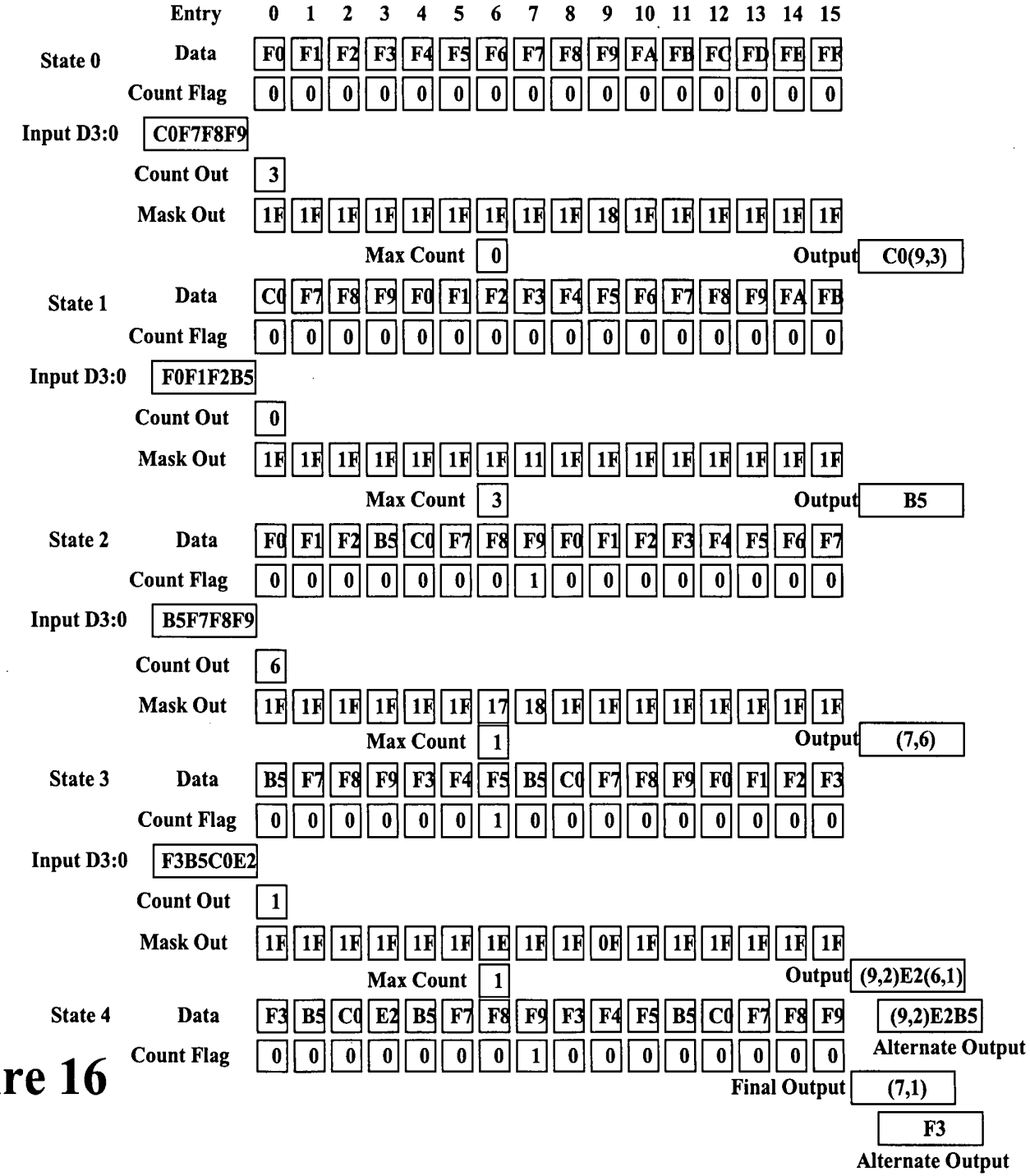


Figure 16

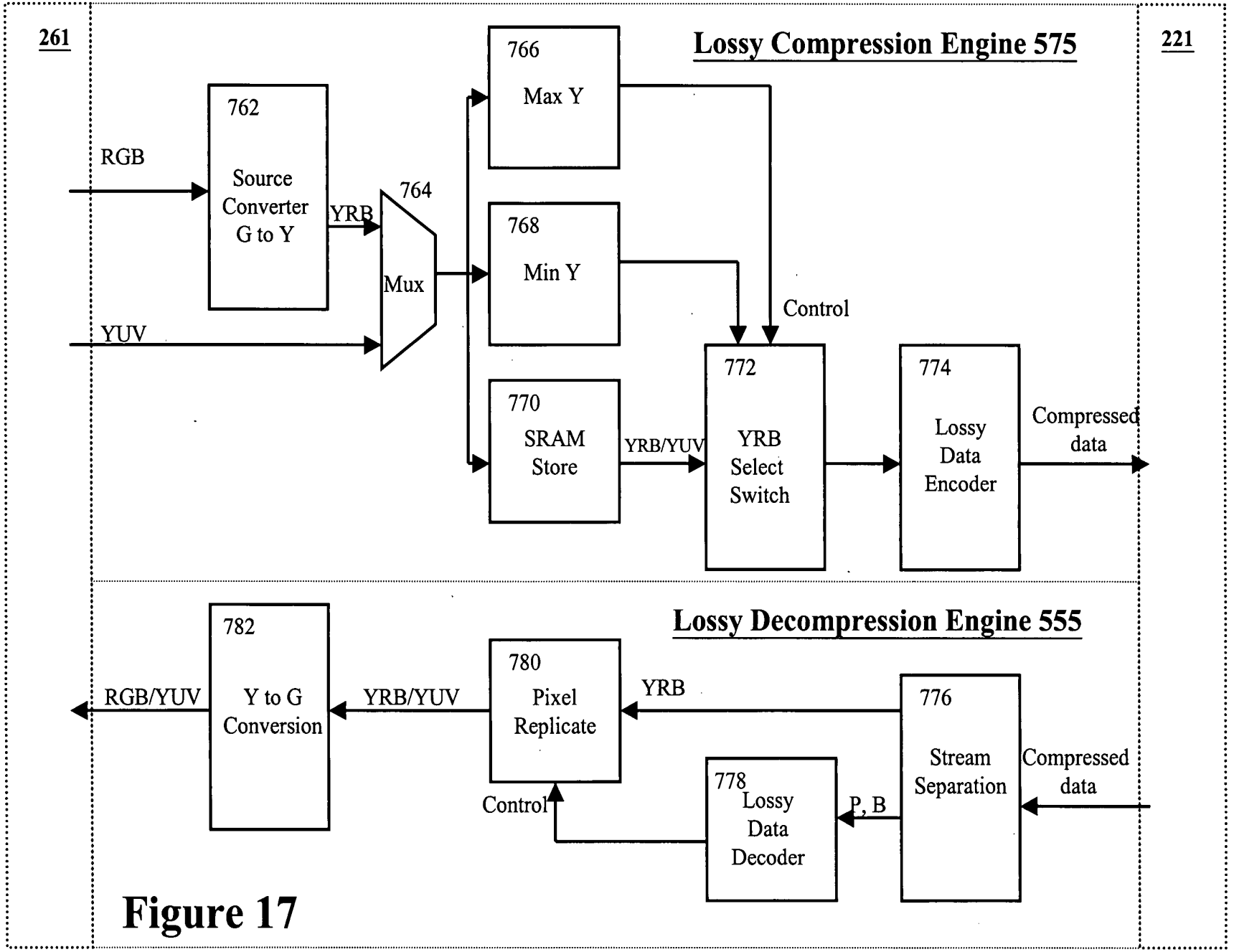

261
221
Lossy Compression Engine 575
766
Max Y
762
Source Converter G to Y
RGB
YRB
764
Mux
768
Min Y
YUV
Control
772
YRB Select Switch
774
Lossy Data Encoder
Compressed data
770
SRAM Store
YRB/YUV
Lossy Decompression Engine 555
782
Y to G Conversion
RGB/YUV
YRB/YUV
780
Pixel Replicate
YRB
776
Stream Separation
Compressed data
778
Lossy Data Decoder
Control
P, B

Figure 17

Ymax = Ymin	1 color	Ymax	Ymax	Rmax	Bmax	11			3 Bytes
		6 bits	6 bits	5 bits	5 bits	2 bits			
Ymax != Ymin	2 colors	Ymax	Ymin	Rmax	Rmin	Bmax	Bmin	P bits	6 Bytes
		6 bits	6 bits	5 bits	5 bits	5 bits	5 bits	16 bits	
Ymax != Ymin	>2 colors	Ymin	Ymax	Rmax	Rmin	Bmax	Bmin	P bits	8 Bytes
		6 bits	6 bits	5 bits	5 bits	5 bits	5 bits	32 bits	

Figure 18

Ymax = Ymin	Amax = Amin = 0x00	1 color	Ymax	Ymax	Rmax	Bmax	00					3 Bytes
			6 bits	6 bits	5 bits	5 bits	2 bits					
Ymax = Ymin	Amax = Amin = 0xFF	1 color	Ymax	Ymax	Rmax	Bmax	11					3 Bytes
			6 bits	6 bits	5 bits	5 bits	2 bits					
Ymax = Ymin	Amax = Amin != 00 or FF	1 color	Ymax	Ymax	Rmax	Bmax	01	Amax	Amin			4/5 Bytes
			6 bits	6 bits	5 bits	5 bits	2 bits	4/8 bits	4/8 bits			
Ymax = Ymin	Amax != Amin	1 color	Ymax	Ymax	Rmax	Bmax	01	Amax	Amin	P bits		6/7 Bytes
		2 Alphas	6 bits	6 bits	5 bits	5 bits	2 bits	4/8 bits	4/8 bits	16 bits		
Ymax = Ymin	Amax != Amin	1 color	Ymax	Ymax	Rmax	Bmax	10	Amax	Amin	P bits		8/9 Bytes
		>2 Alphas	6 bits	6 bits	5 bits	5 bits	2 bits	4/8 bits	4/8 bits	32 bits		
Ymax != Ymin	X	2 colors	Ymax	Ymin	Rmax	Rmin	Bmax	Bmin	Amax	Amin	P bits	7/8 Bytes
			6 bits	6 bits	5 bits	5 bits	5 bits	5 bits	4/8 bits	4/8 bits	16 bits	
Ymax != Ymin	X	>2 colors	Ymin	Ymax	Rmax	Rmin	Bmax	Bmin	Amax	Amin	P bits	9/10 Bytes
			6 bits	6 bits	5 bits	5 bits	5 bits	5 bits	4/8 bits	4/8 bits	32 bits	

Figure 19

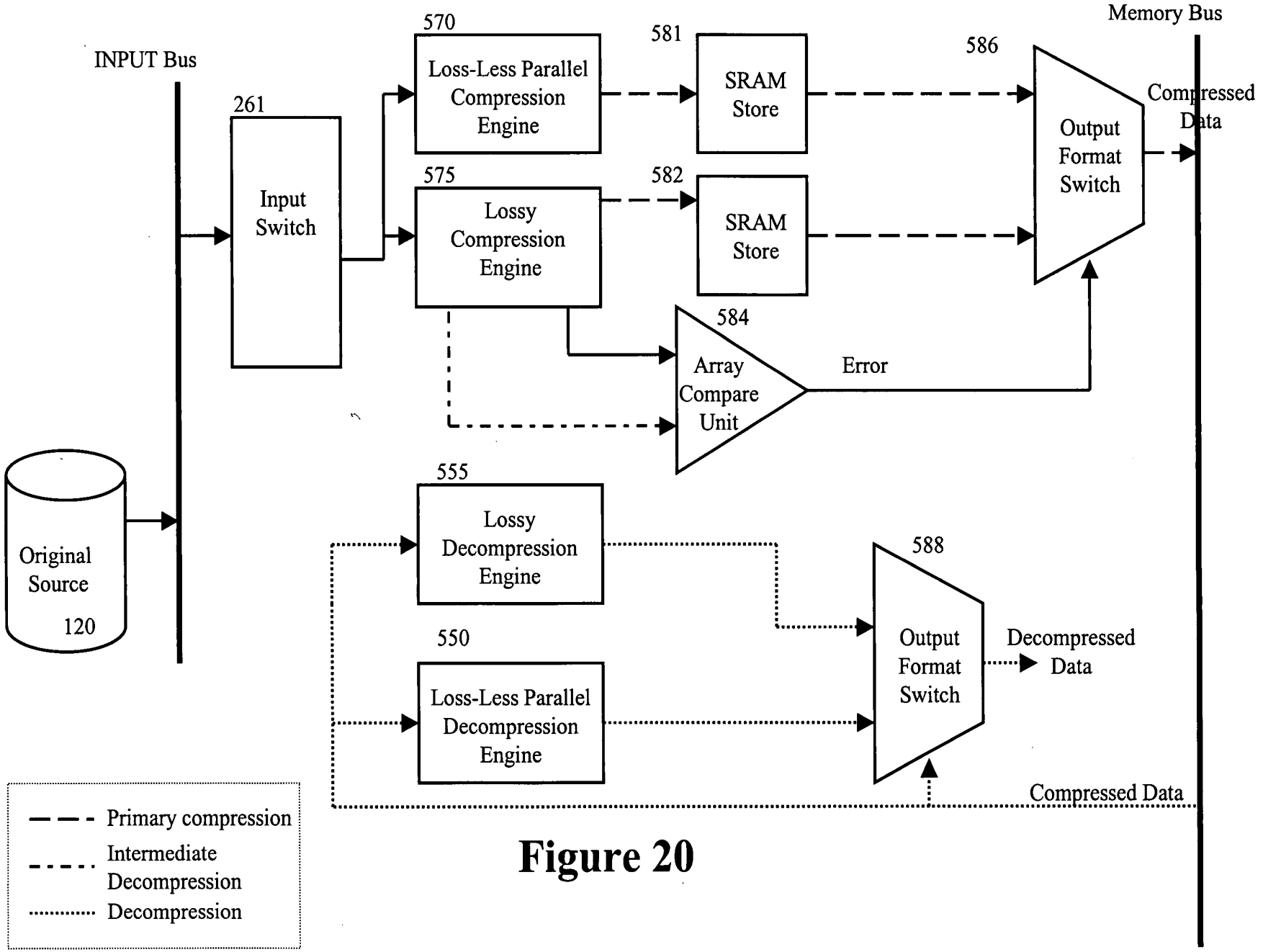

INPUT Bus
261
Input Switch
570
Loss-Less Parallel Compression Engine
575
Lossy Compression Engine
581
SRAM Store
582
SRAM Store
584
Array Compare Unit
Error
586
Output Format Switch
Memory Bus
Compressed Data
Original Source
120
555
Lossy Decompression Engine
550
Loss-Less Parallel Decompression Engine
588
Output Format Switch
Decompressed Data
Compressed Data
Primary compression
Intermediate Decompression
Decompression

Figure 20

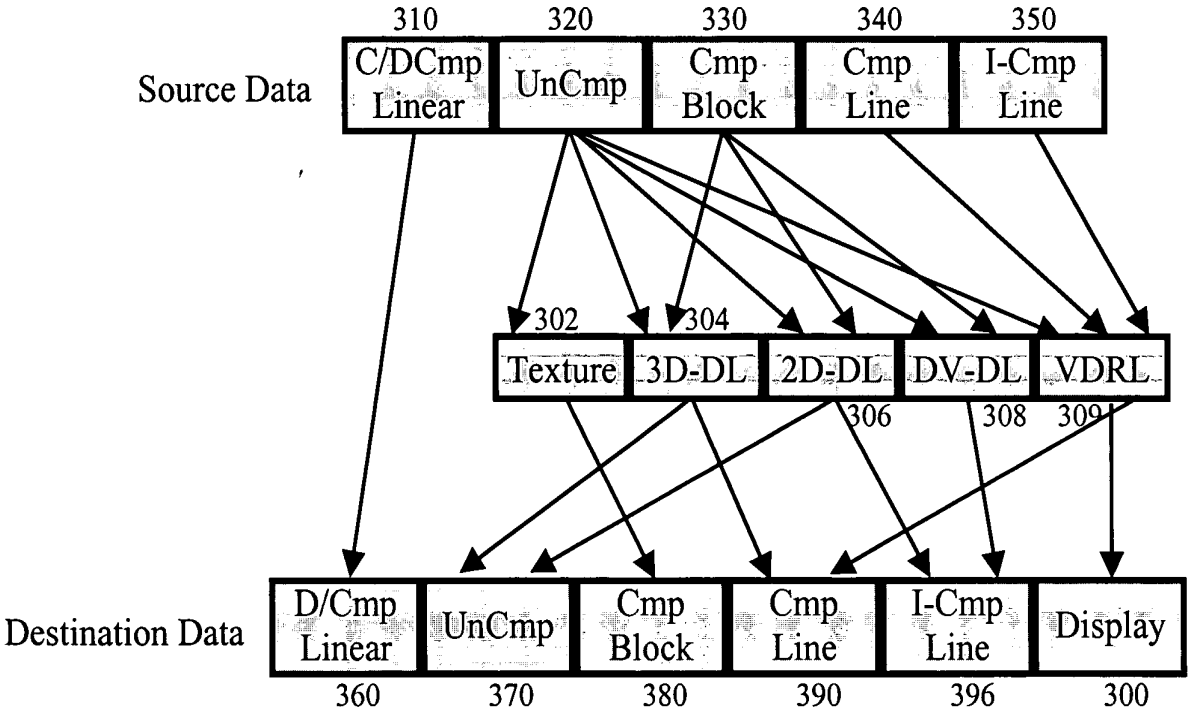

310
320
330
340
350
Source Data
C/DCmp Linear
UnCmp
Cmp Block
Cmp Line
I-Cmp Line
302
304
Texture
3D-DL
2D-DL
DV-DL
VDRL
306
308
309
Destination Data
D/Cmp Linear
UnCmp
Cmp Block
Cmp Line
I-Cmp Line
Display
360
370
380
390
396
300

Figure 21

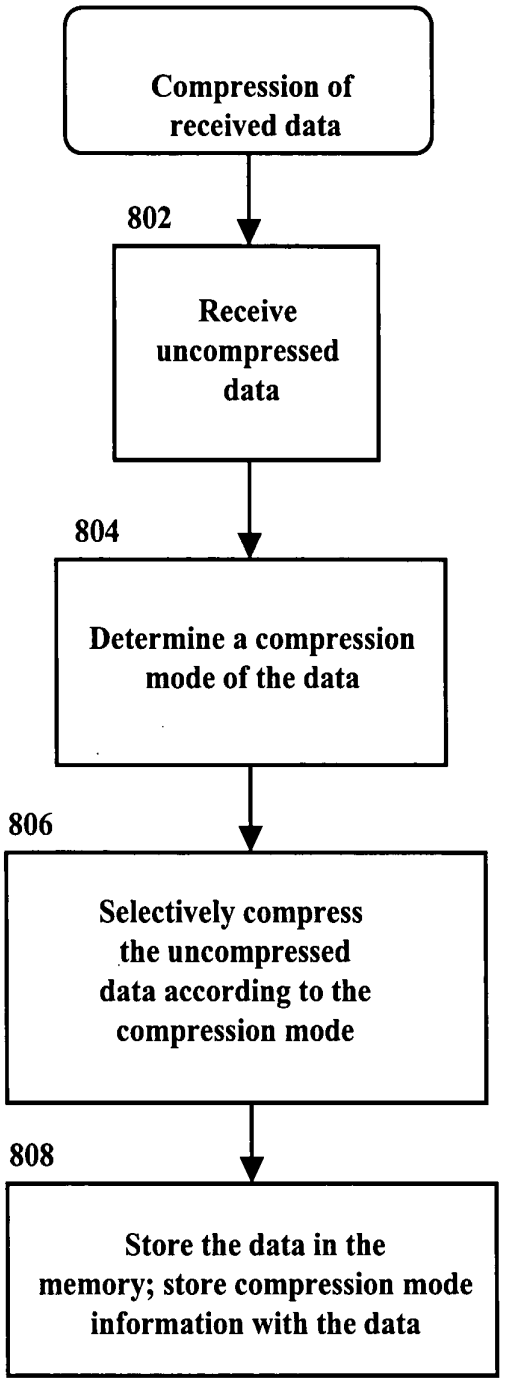

Compression of received data
802
Receive uncompressed data
804
Determine a compression mode of the data
806
Selectively compress the uncompressed data according to the compression mode
808
Store the data in the memory; store compression mode information with the data

Figure 22

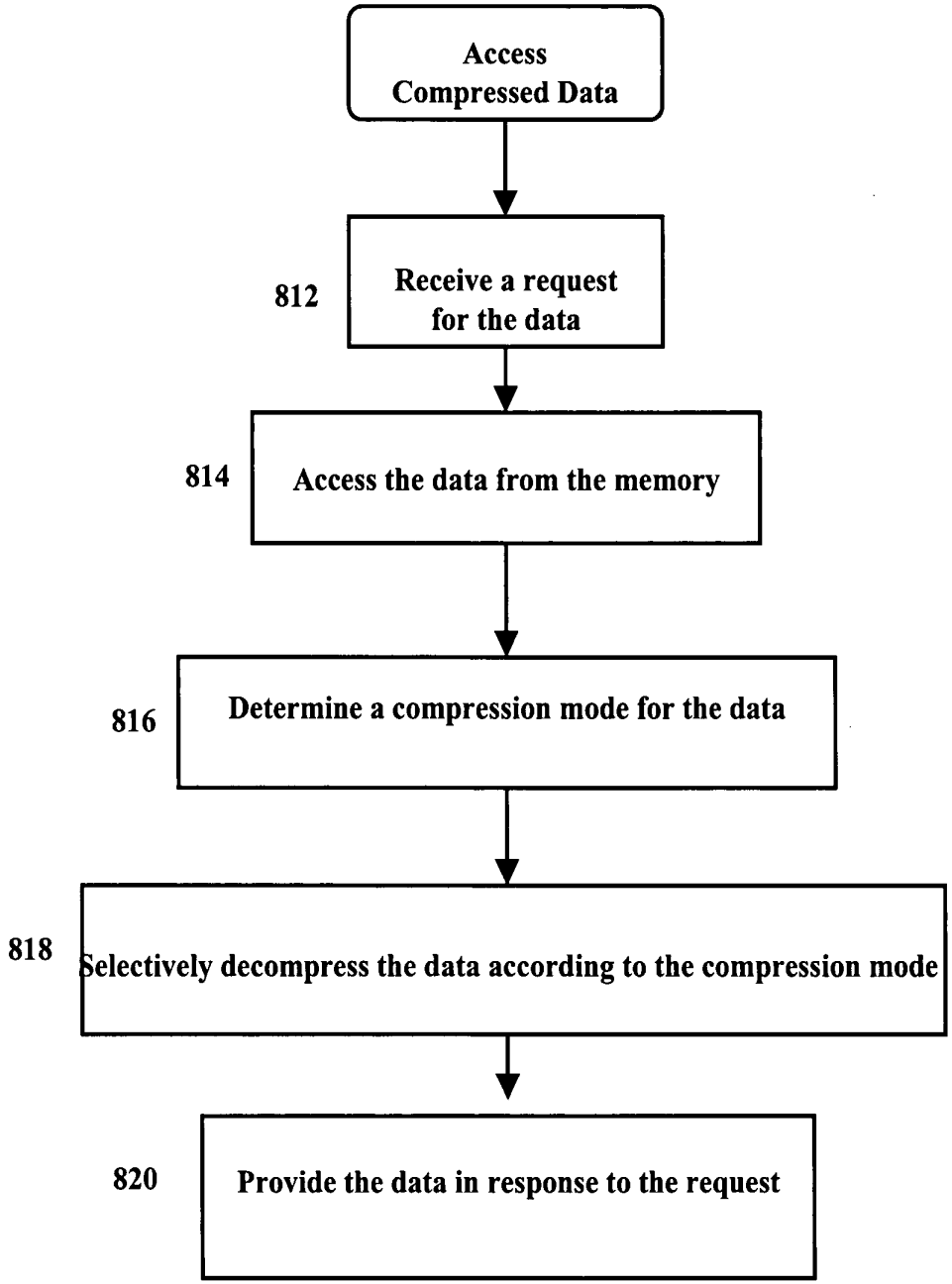

Access Compressed Data
812
Receive a request for the data
814
Access the data from the memory
816
Determine a compression mode for the data
818
Selectively decompress the data according to the compression mode
820
Provide the data in response to the request

Figure 23

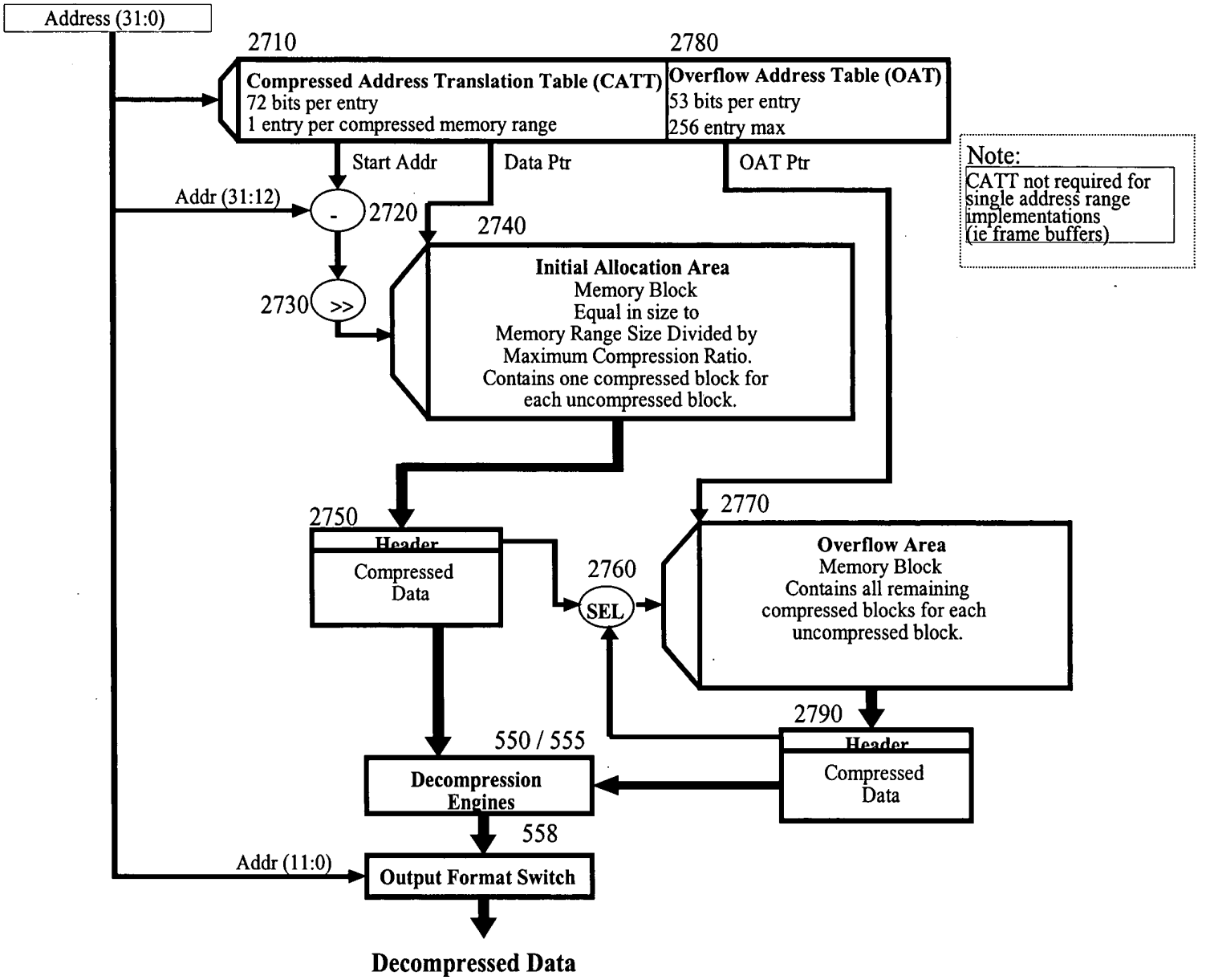

Address (31:0)
2710
2780
Compressed Address Translation Table (CATT)
72 bits per entry
1 entry per compressed memory range
Overflow Address Table (OAT)
53 bits per entry
256 entry max
Start Addr
Data Ptr
OAT Ptr
Note:
CATT not required for single address range implementations (ie frame buffers)
Addr (31:12)
-
2720
2740
2730
>>
Initial Allocation Area
Memory Block
Equal in size to
Memory Range Size Divided by
Maximum Compression Ratio.
Contains one compressed block for each uncompressed block.
2750
2770
Header
Compressed Data
2760
SEL
Overflow Area
Memory Block
Contains all remaining compressed blocks for each uncompressed block.
2790
550 / 555
Header
Compressed Data
Decompression Engines
558
Addr (11:0)
Output Format Switch
Decompressed Data

Figure 24

Compressed Address Translation Table (CATT) – 128 Entry Design Limit				
Starting Addr	Ending Addr	Type	Data Ptr	OAT Ptr
20 bits	20 bits	4 bits	20 bits	8 bits
4GB Addressability		Compressed		
4K Boundry	4K Boundry	Blk Size	4K Boundry	4K Boundry

Overflow Address Table (OAT) – 256 Entry Max			
Overflow Ptr	Next Block Ptr	Next OAT Ptr	Next OAT Valid
20 bits	24 bits	8 bits	1 bit
4 GB Addressability		Points to next entry	
4K Boundry		in this table	

Initial Header Description		
Value	# of bits	Meaning
0	1	Last Block/Unused
10 A (20 bits)	22	The next block is at offset A in the Overflow Area
11 IA(8+20 bits)	30	The next block is at offset A in the Overflow Area of OAT entry I

Overflow Header Description		
Value	# of bits	Meaning
00	2	Last Block/Unused
01	2	The next block follows physically after this one
10A (8 bits)	10	The next block is A blocks before this one (or after?)
110A (20 bits)	23	The next block is at offset A in the Overflow Area
111 IA (8+20 bits)	31	The next block is at offset A in the Overflow Area of OAT entry I

Figure 25 - Memory Allocation Fields

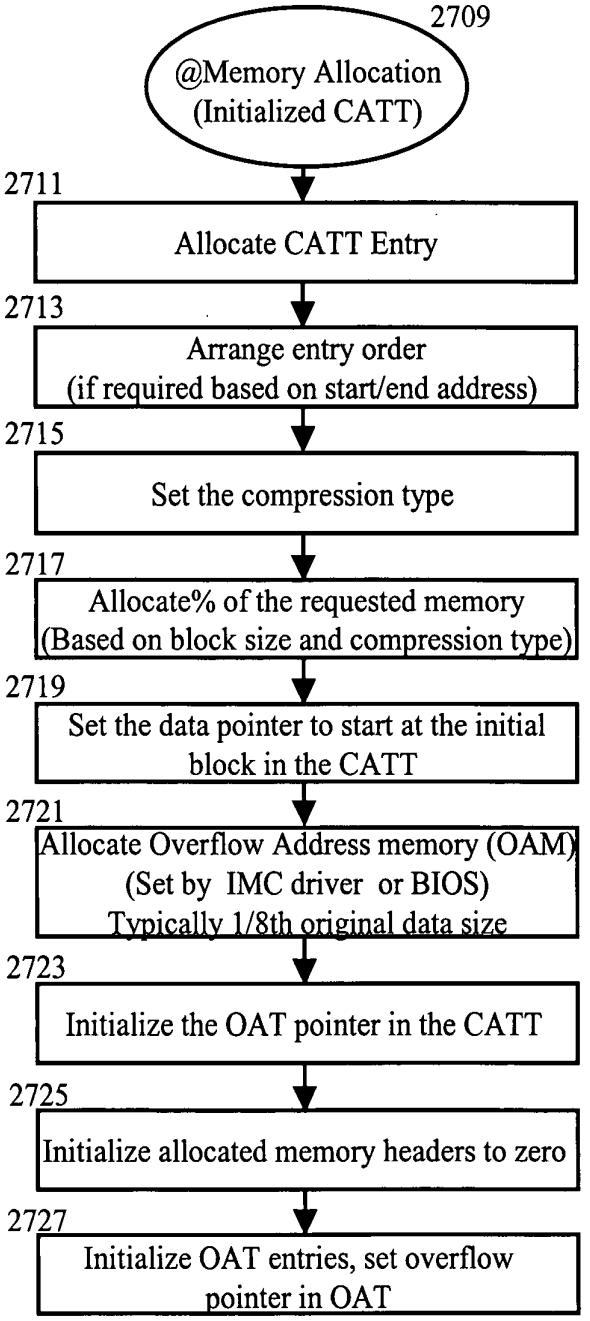

2709
@Memory Allocation
(Initialized CATT)
2711
Allocate CATT Entry
2713
Arrange entry order
(if required based on start/end address)
2715
Set the compression type
2717
Allocate% of the requested memory
(Based on block size and compression type)
2719
Set the data pointer to start at the initial
block in the CATT
2721
Allocate Overflow Address memory (OAM)
(Set by IMC driver or BIOS)
Typically 1/8th original data size
2723
Initialize the OAT pointer in the CATT
2725
Initialize allocated memory headers to zero
2727
Initialize OAT entries, set overflow
pointer in OAT

Figure 26

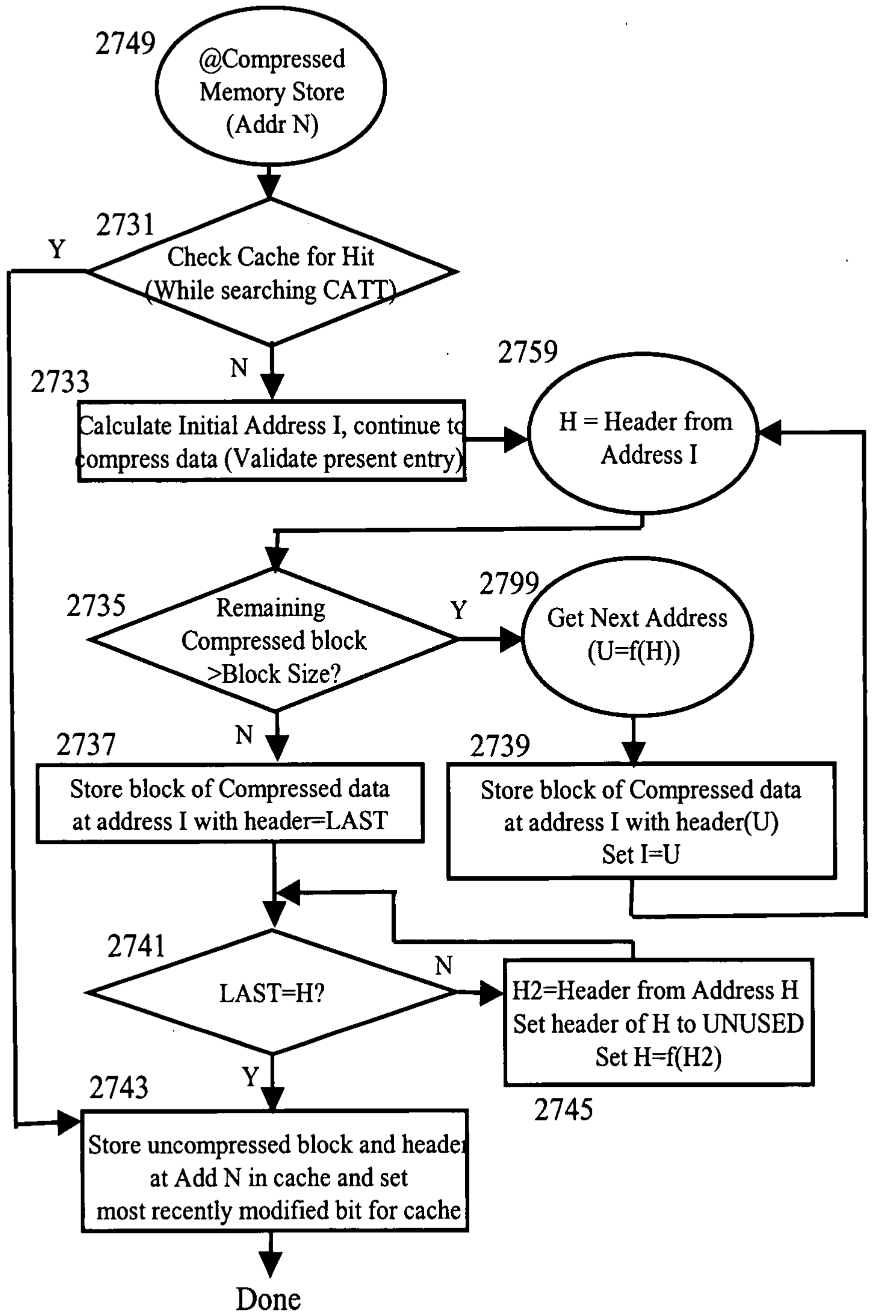

2749
@Compressed
Memory Store
(Addr N)
2731
Check Cache for Hit
(While searching CATT)
Y
N
2733
Calculate Initial Address I, continue to
compress data (Validate present entry)
2759
H = Header from
Address I
2735
Remaining
Compressed block
>Block Size?
Y
2799
Get Next Address
(U=f(H))
N
2737
Store block of Compressed data
at address I with header=LAST
2739
Store block of Compressed data
at address I with header(U)
Set I=U
2741
LAST=H?
N
H2=Header from Address H
Set header of H to UNUSED
Set H=f(H2)
2745
Y
2743
Store uncompressed block and header
at Add N in cache and set
most recently modified bit for cache
Done

Figure 27

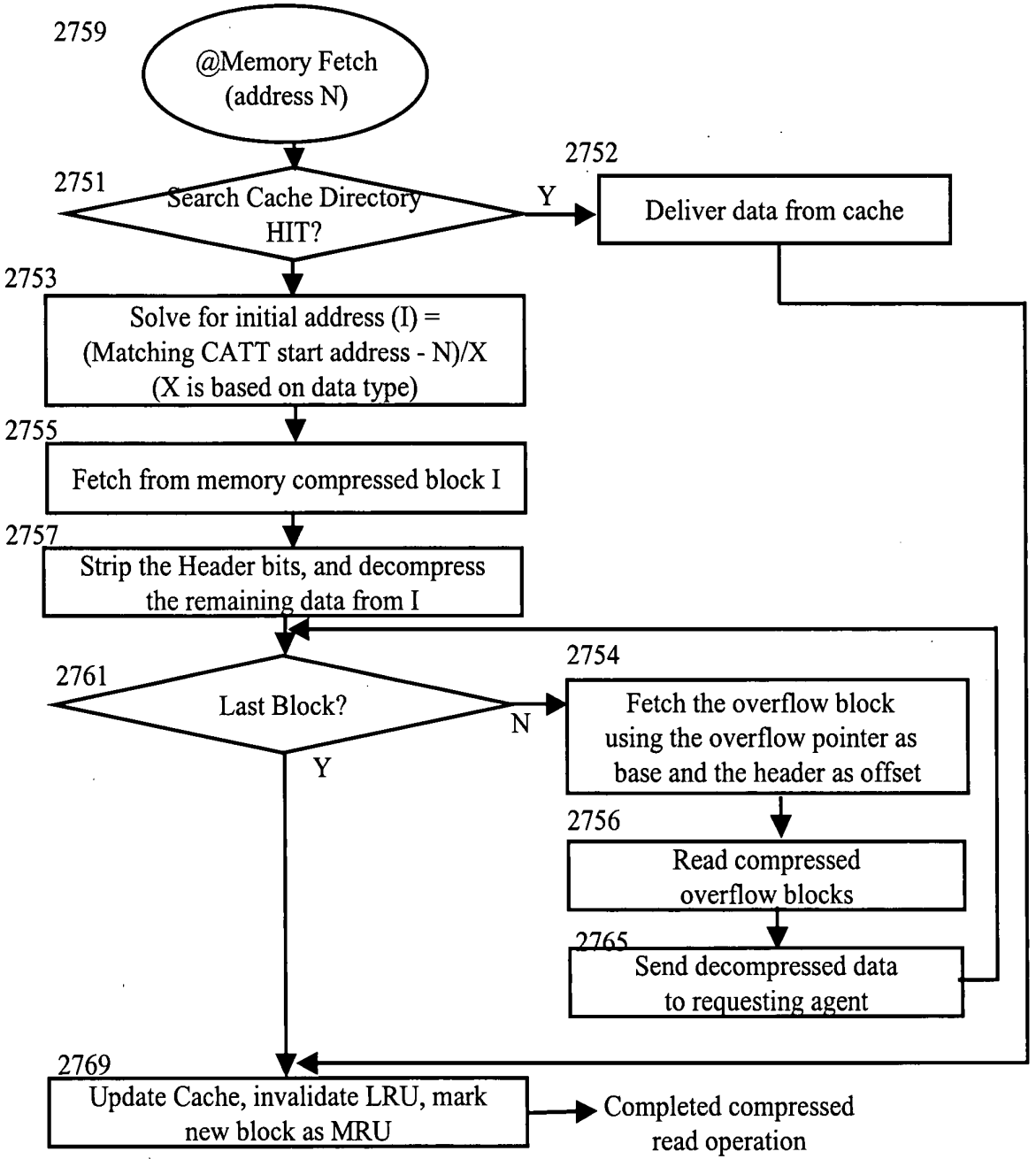

2759
@Memory Fetch
(address N)
2751
Search Cache Directory
HIT?
Y
2752
Deliver data from cache
2753
Solve for initial address (I) =
(Matching CATT start address - N)/X
(X is based on data type)
2755
Fetch from memory compressed block I
2757
Strip the Header bits, and decompress
the remaining data from I
2761
Last Block?
N
Y
2754
Fetch the overflow block
using the overflow pointer as
base and the header as offset
2756
Read compressed
overflow blocks
2765
Send decompressed data
to requesting agent
2769
Update Cache, invalidate LRU, mark
new block as MRU
Completed compressed
read operation

Figure 28

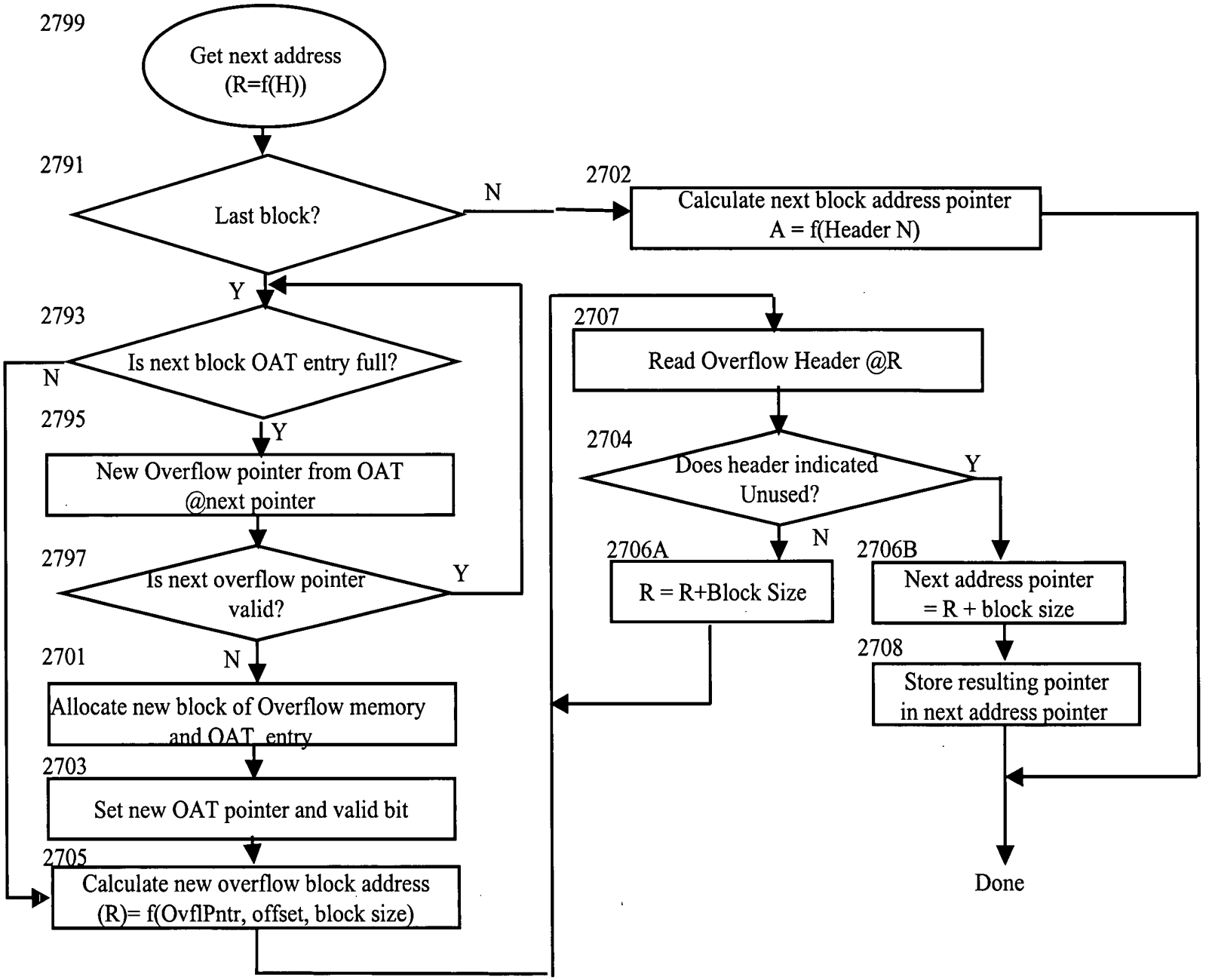
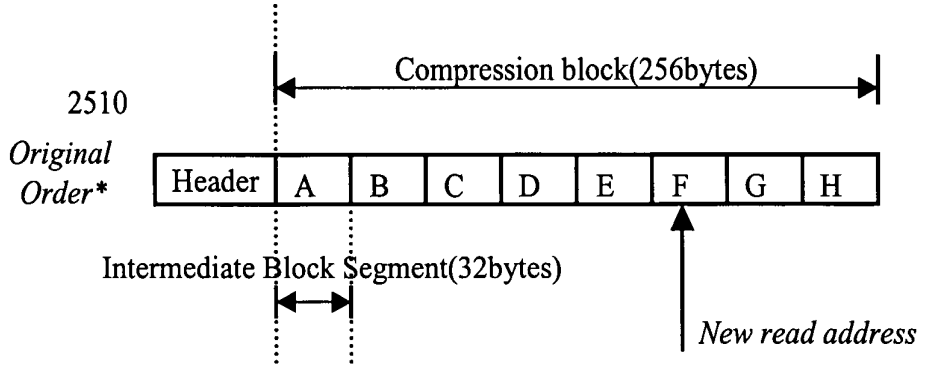

2799
Get next address (R=f(H))
2791
Last block?
N
2702
Calculate next block address pointer A = f(Header N)
Y
2793
Is next block OAT entry full?
N
Y
2795
New Overflow pointer from OAT @next pointer
2797
Is next overflow pointer valid?
Y
N
2701
Allocate new block of Overflow memory and OAT entry
2703
Set new OAT pointer and valid bit
2705
Calculate new overflow block address (R)= f(OvflPntr, offset, block size)
2707
Read Overflow Header @R
2704
Does header indicated Unused?
Y
N
2706A
R = R+Block Size
2706B
Next address pointer = R + block size
2708
Store resulting pointer in next address pointer
Done

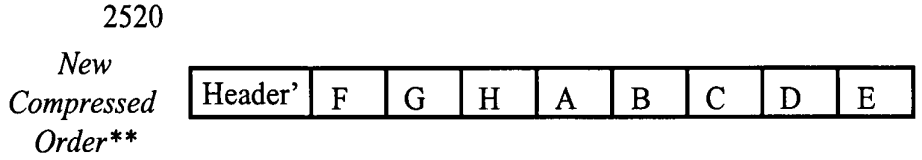
Figure 29

Uncomp Block Bytes	Type	Initial Block Size Bytes	Overflow Block Size Bytes	Max Comp Ratio (X:1)	Initial Allocation	Header w/o OF	Header w/ OF Non-Frag	Header w/ OF Fragmented
4096	8	256	64	16	6%	0.0%	0.4%	4.1%
2048	7	128	64	16	6%	0.1%	0.5%	4.2%
1024	6	64	64	16	6%	0.2%	0.6%	4.3%
512	5	64	64	8	13%	0.2%	0.9%	4.3%
256	4	64	64	4	25%	0.2%	1.4%	4.3%
128	3	32	32	4	25%	0.4%	2.8%	8.8%
64	2	32	16	2	50%	0.4%	5.1%	13.6%
32	1	32	8	1	100%	0.4%	8.9%	11.5%

Figure 30



* Header tag field is '0' indicating in original order



** Header tag field is '5' indicating out of order

2530

1) Read compressed block from memory
2) Decompress to L3 data cache
3) Store Position tag (@F) in table
4) Re-compress block out of order
5) Attach header with tag (@F)
6) Write compressed block back to memory

Figure 31

0092T0" EhET6h60

Bytes Compressed	Flag	Index	Count	Data	**Bits Used**
0	0	-	-	8b	9
1	10	6b	-	-	8
2	1100	6b	-	-	10
3	1101	6b	-	-	10
4	1110	6b	-	-	10
5	1111000	6b	-	-	13
6	1111001	6b	-	-	13
7	1111010	6b	-	-	13
8	1111011	6b	-	-	13
9	1111100	6b	-	-	13
10	1111101	6b	-	-	13
11	1111110	6b	-	-	13
>11	1111111	6b	12b	-	25

Figure 32

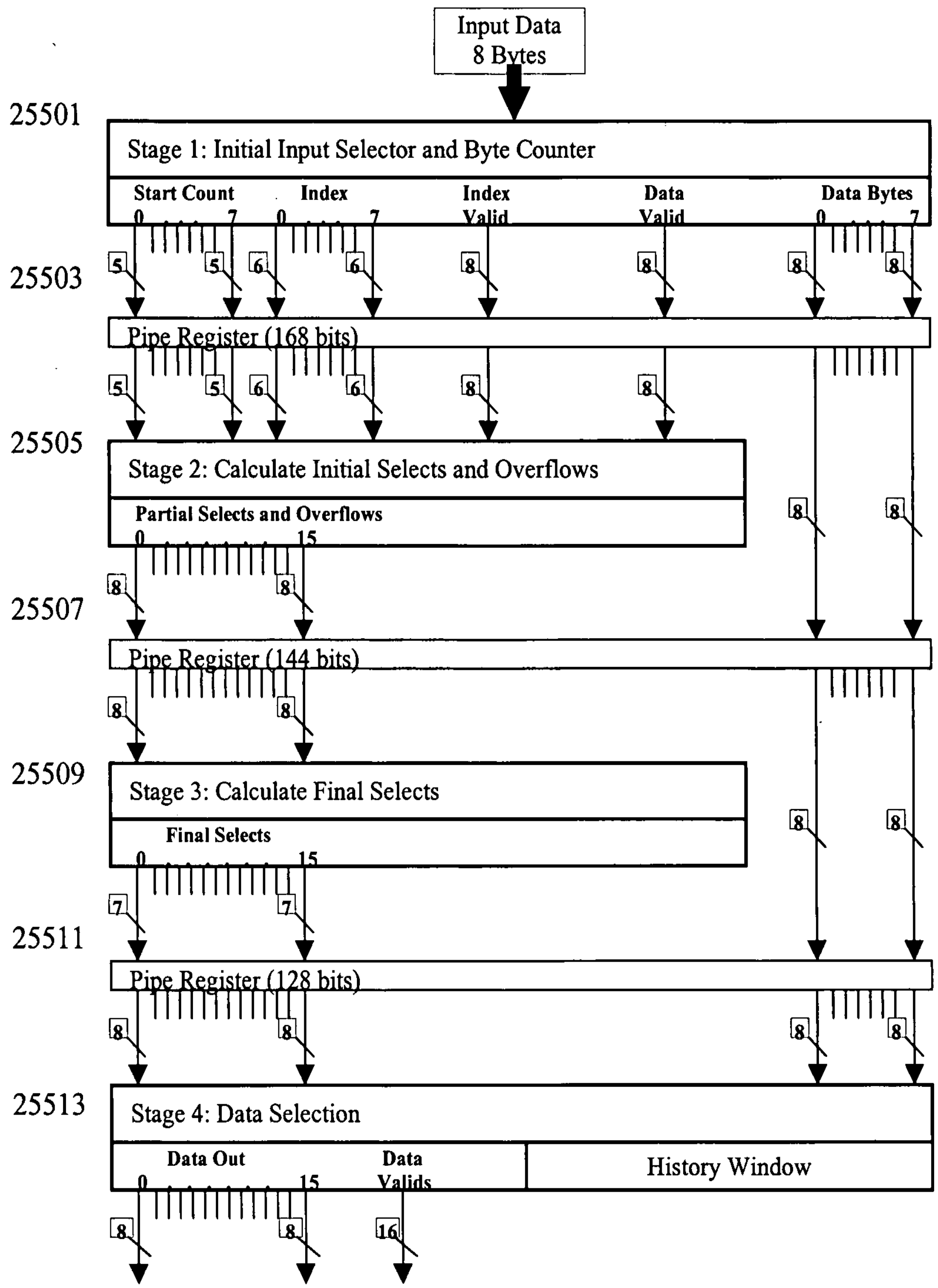

Input Data
8 Bytes
25501
Stage 1: Initial Input Selector and Byte Counter
Start Count
Index
Index Valid
Data Valid
Data Bytes
25503
Pipe Register (168 bits)
25505
Stage 2: Calculate Initial Selects and Overflows
Partial Selects and Overflows
25507
Pipe Register (144 bits)
25509
Stage 3: Calculate Final Selects
Final Selects
25511
Pipe Register (128 bits)
25513
Stage 4: Data Selection
Data Out
Data Valids
History Window

Figure 33

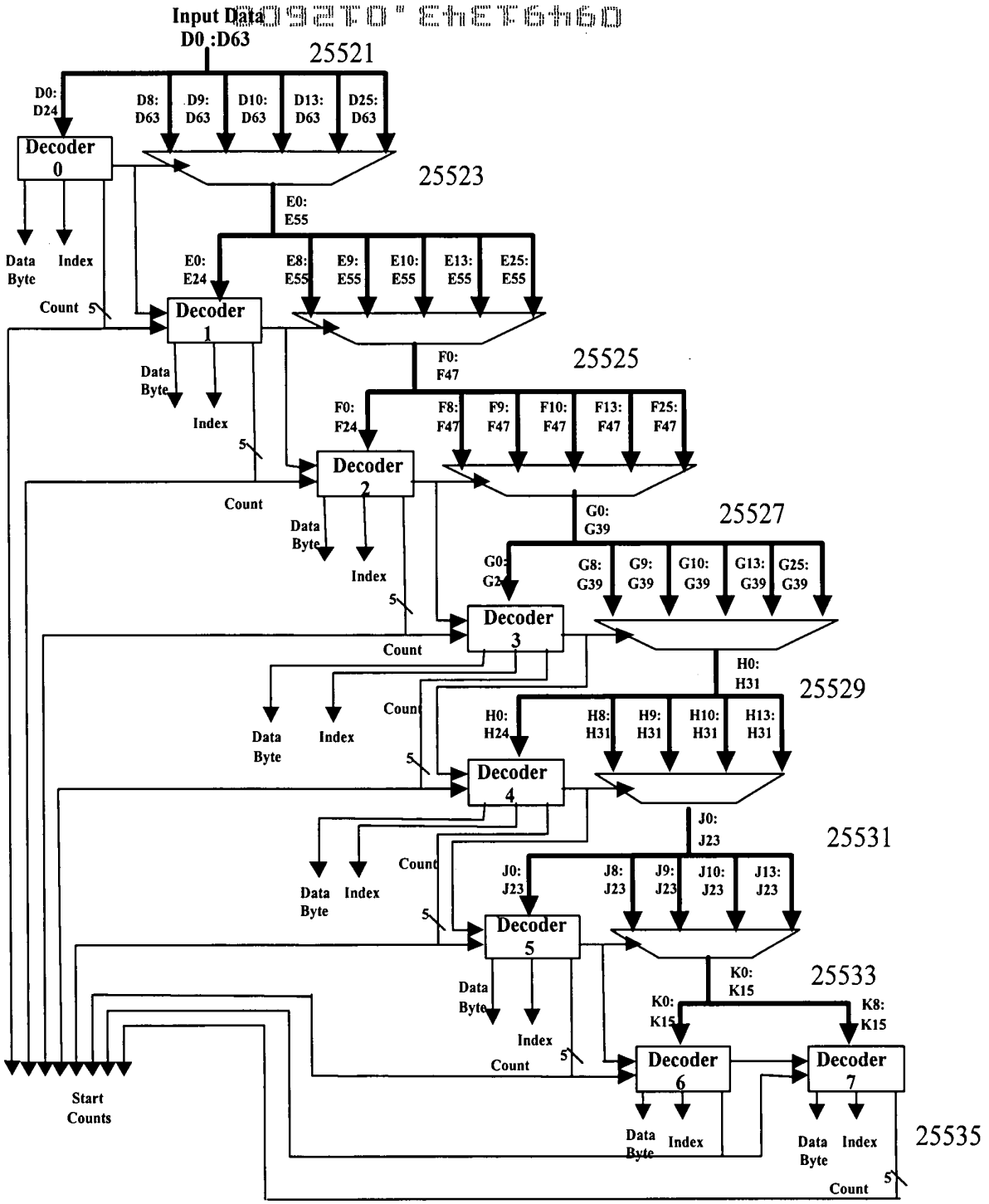

Input Data
D0 :D63
25521
D0: D24
D8: D63
D9: D63
D10: D63
D13: D63
D25: D63
Decoder 0
25523
E0: E55
Data Byte
Index
E0: E24
E8: E55
E9: E55
E10: E55
E13: E55
E25: E55
Count 5
Decoder 1
F0: F47
25525
Data Byte
Index
F0: F24
F8: F47
F9: F47
F10: F47
F13: F47
F25: F47
5
Decoder 2
Count
G0: G39
25527
Data Byte
Index
G0: G24
G8: G39
G9: G39
G10: G39
G13: G39
G25: G39
5
Decoder 3
Count
H0: H31
25529
Data Byte
Index
Count
H0: H24
H8: H31
H9: H31
H10: H31
H13: H31
5
Decoder 4
J0: J23
25531
Data Byte
Index
Count
J0: J23
J8: J23
J9: J23
J10: J23
J13: J23
5
Decoder 5
K0: K15
25533
Data Byte
Index
Count
K0: K15
K8: K15
5
Decoder 6
Decoder 7
Start Counts
Data Byte
Index
Data Byte
Index
25535
Count
5

Figure 34

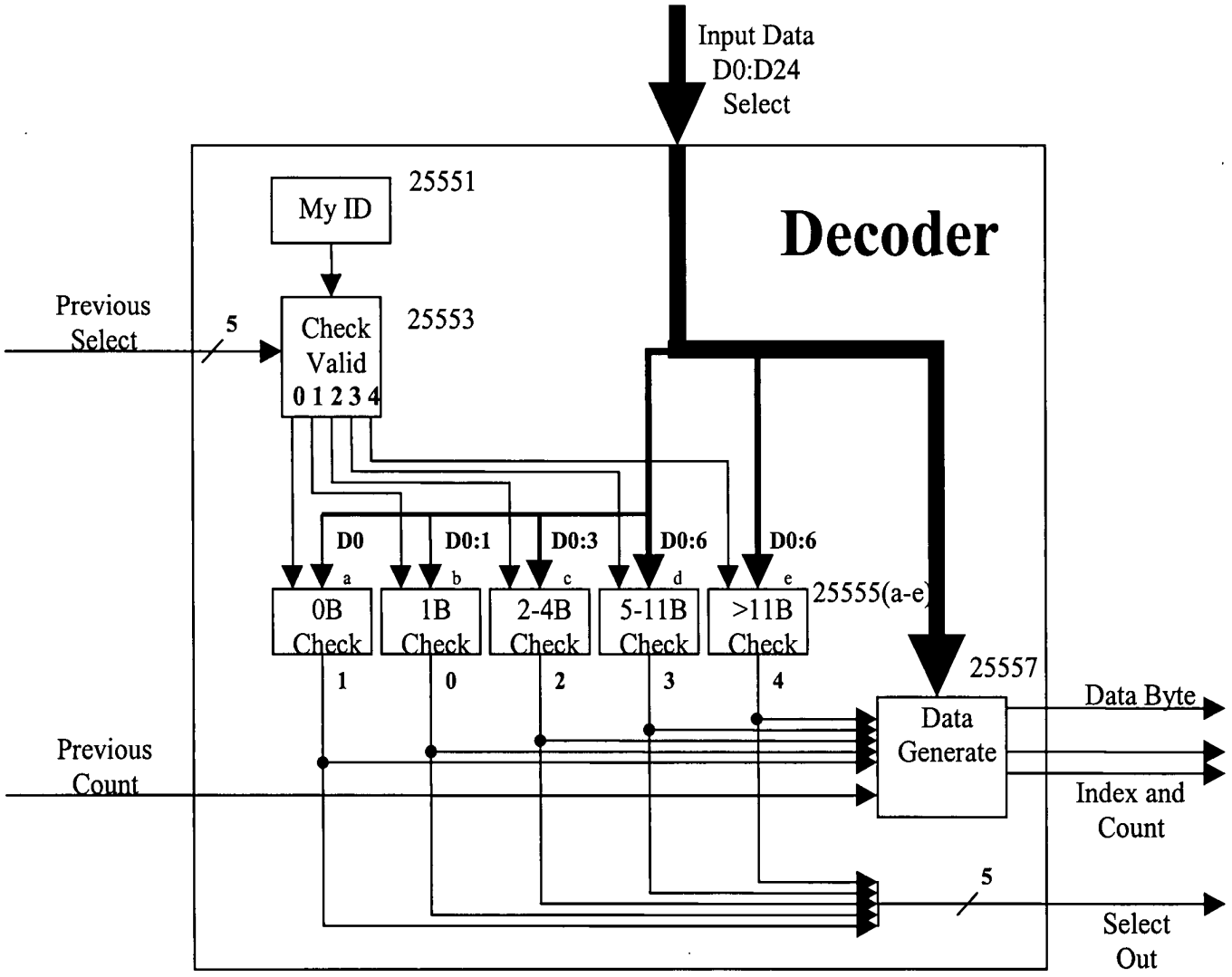

Input Data
D0:D24
Select
Decoder
My ID
25551
Previous
Select
5
Check
Valid
0 1 2 3 4
25553
D0
D0:1
D0:3
D0:6
D0:6
a
b
c
d
e
0B
Check
1B
Check
2-4B
Check
5-11B
Check
>11B
Check
25555(a-e)
1
0
2
3
4
25557
Data
Generate
Data Byte
Index and
Count
Previous
Count
5
Select
Out

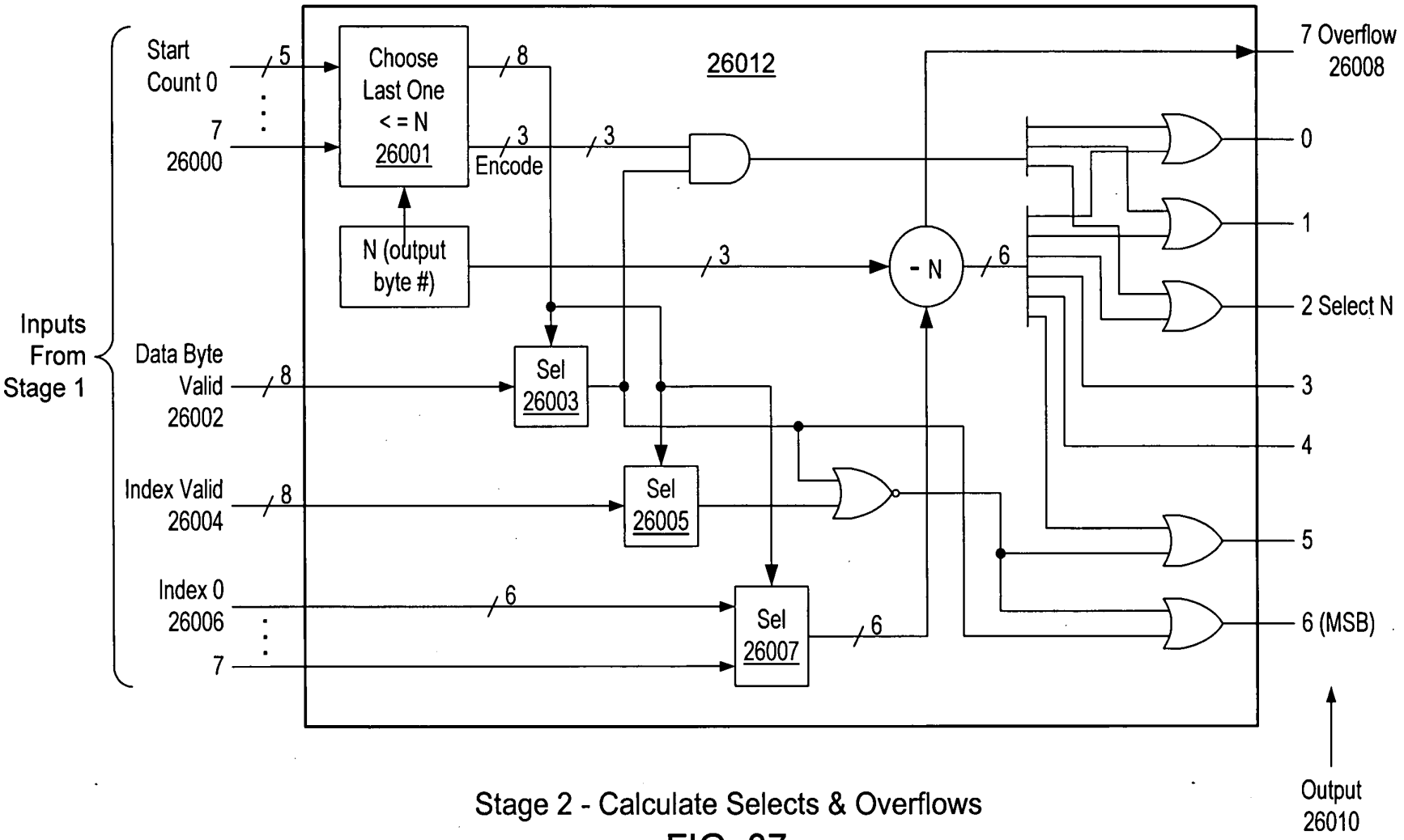
Figure 35

Previous Select	10	08	04	02	01	00
My ID=01	1F	1F	1F	1F	1F	00
My ID=02	1F	1F	1F	1F	1F	00
My ID=04	1F	1F	1F	1F	1F	00
My ID=08	1F	1F	1F	1F	1E	00
My ID=10	1F	1F	1F	1F	1E	00
My ID=20	1E	1E	1E	1E	00	00
My ID=40	1E	1E	1E	1C	00	00
My ID=80	08	00	00	00	00	00

Figure 36a

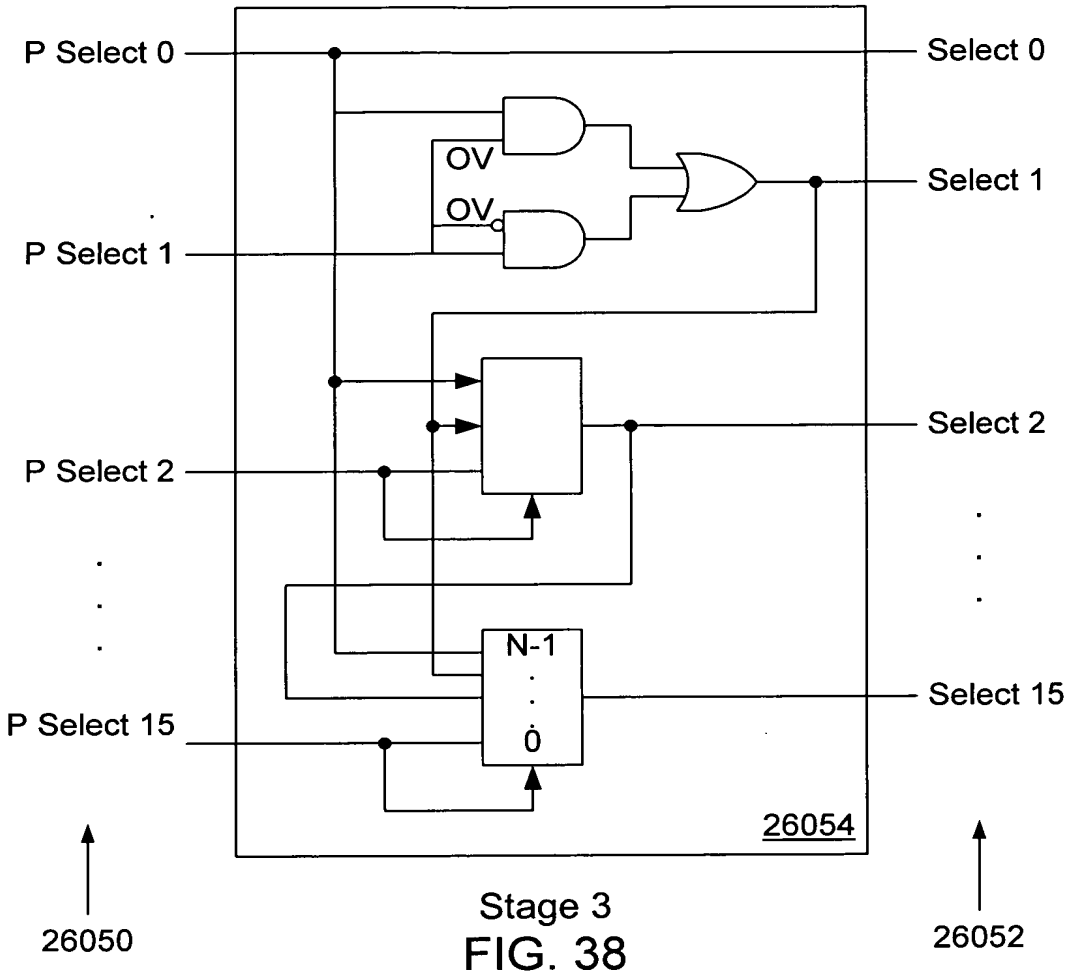
Select	10	08	04	02	01	00
Data Byte	X	D1:D8	X	X	X	X
Index	D2:D7	X	D4:D9	D7:D12	D7:D12	X
Count	PC+1	PC+1	D2:D3+PC+2	D4:D6+PC+5	D13:D24+PC	X

Figure 36b



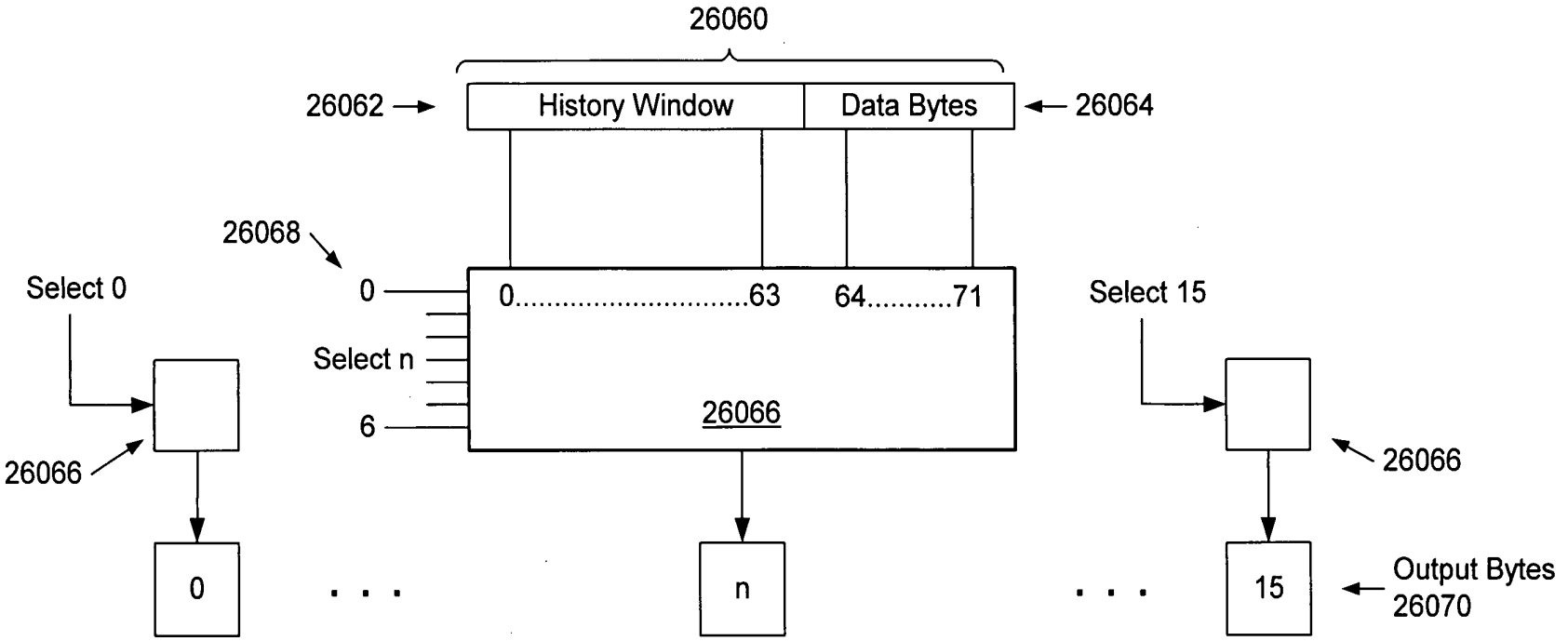
Stage 2 - Calculate Selects & Overflows

FIG. 37



Stage 3

FIG. 38



Stage 4

FIG. 39

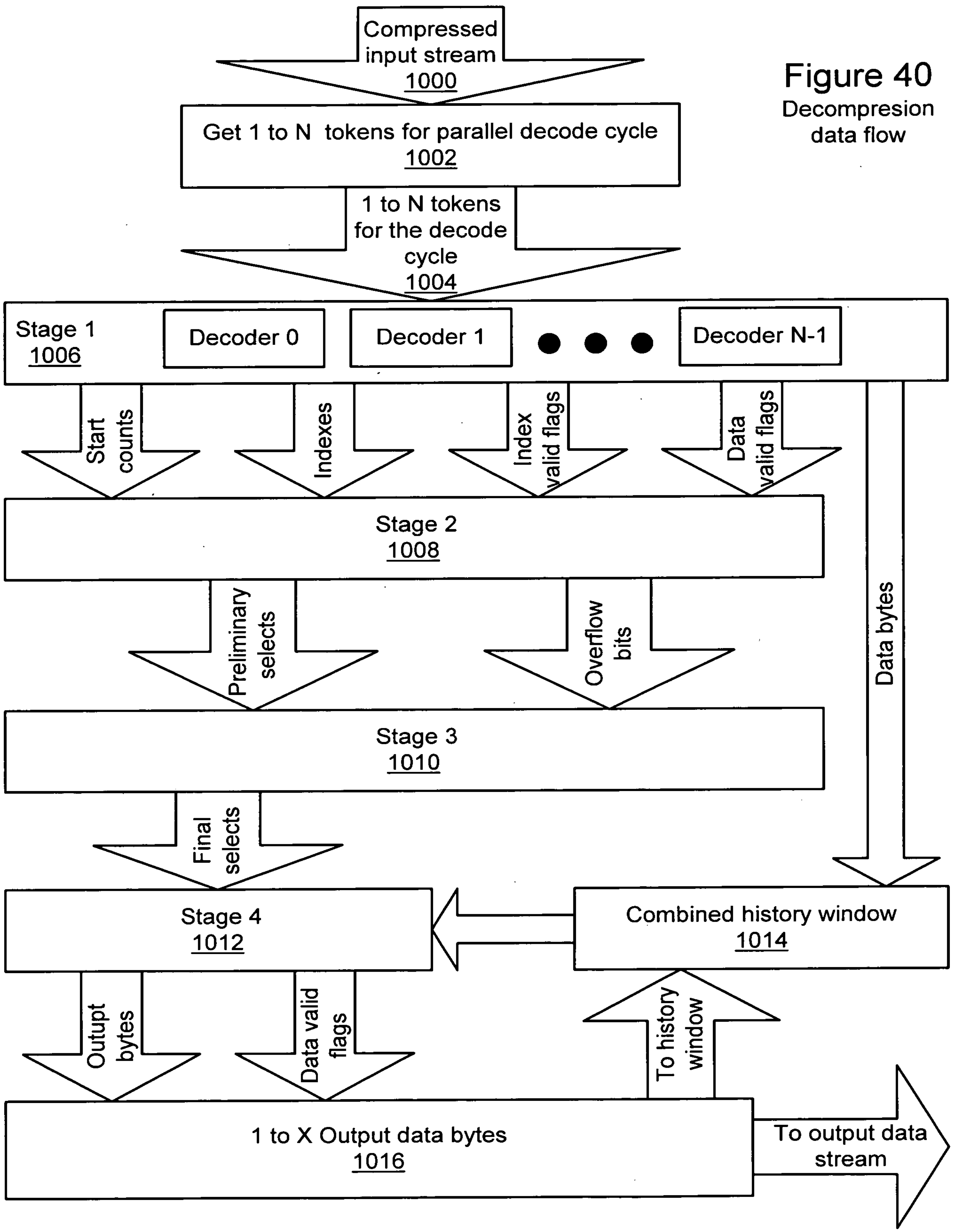


Figure 40
Decompresion data flow

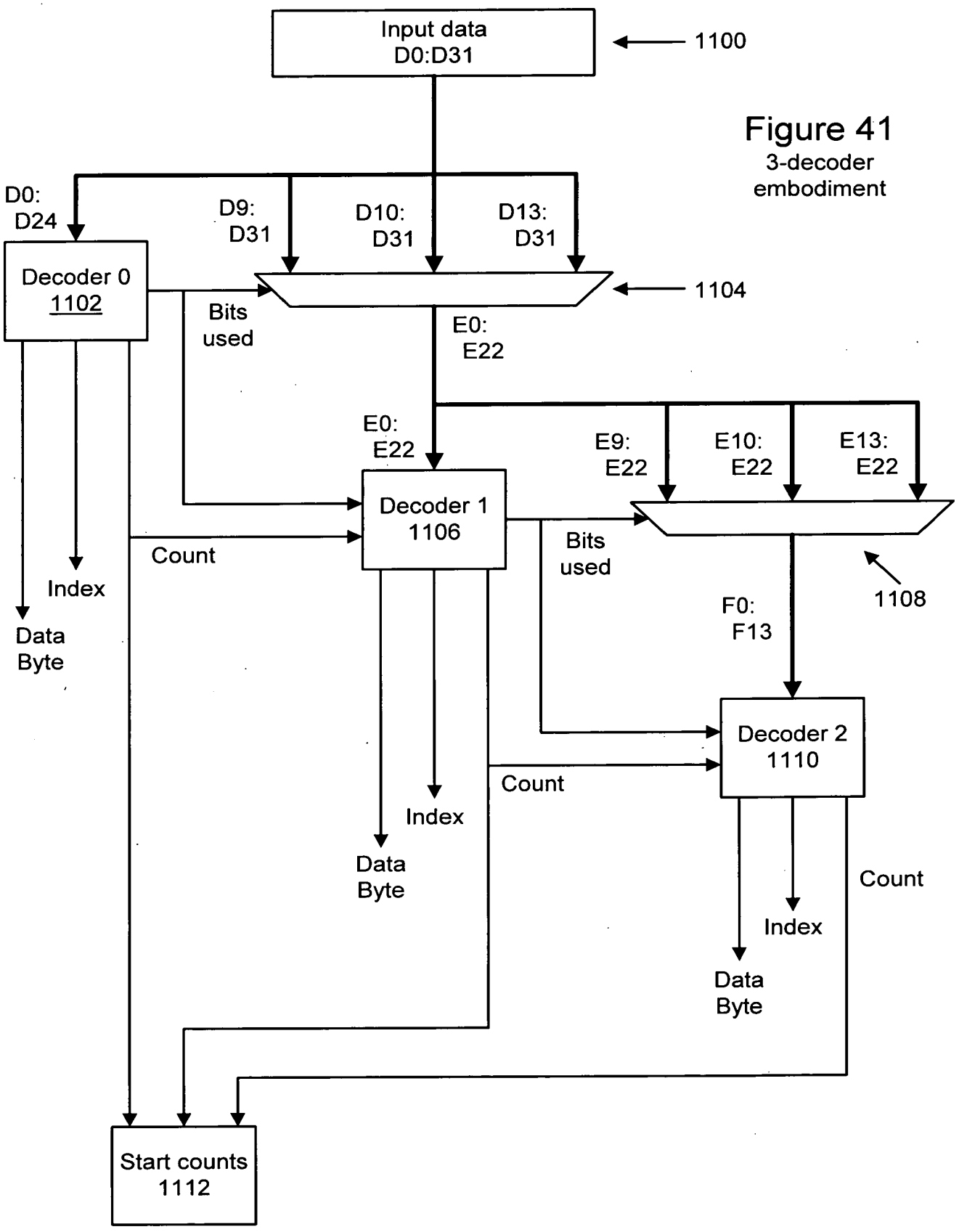


Figure 41
3-decoder
embodiment

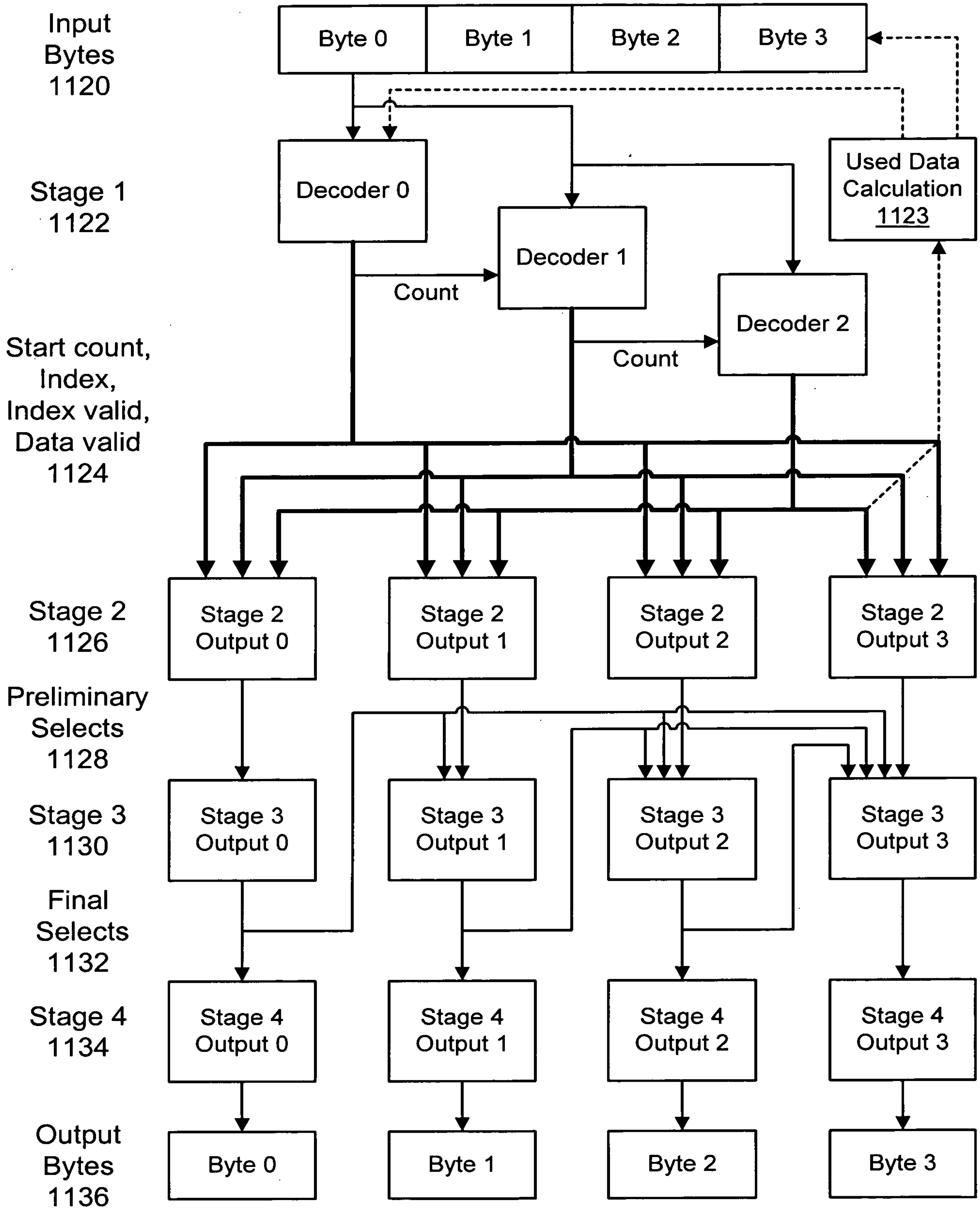

Input Bytes 1120
Byte 0
Byte 1
Byte 2
Byte 3
Stage 1 1122
Decoder 0
Decoder 1
Decoder 2
Used Data Calculation 1123
Count
Count
Start count, Index, Index valid, Data valid 1124
Stage 2 1126
Stage 2 Output 0
Stage 2 Output 1
Stage 2 Output 2
Stage 2 Output 3
Preliminary Selects 1128
Stage 3 1130
Stage 3 Output 0
Stage 3 Output 1
Stage 3 Output 2
Stage 3 Output 3
Final Selects 1132
Stage 4 1134
Stage 4 Output 0
Stage 4 Output 1
Stage 4 Output 2
Stage 4 Output 3
Output Bytes 1136
Byte 0
Byte 1
Byte 2
Byte 3

Figure 42a - 4 input bytes, 3 decoders, 4 output bytes

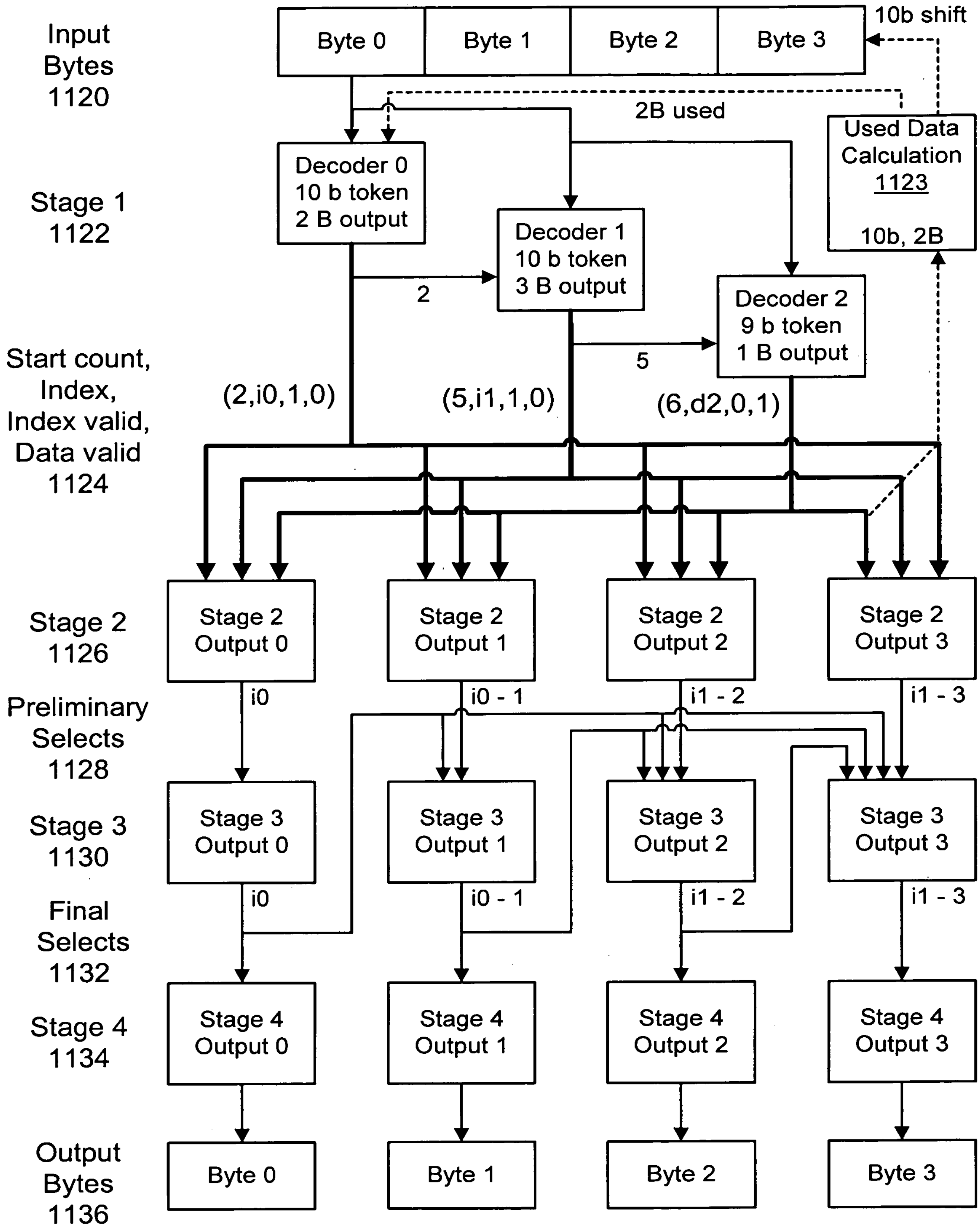

Input Bytes 1120
Byte 0
Byte 1
Byte 2
Byte 3
10b shift
2B used
Used Data Calculation 1123
10b, 2B
Stage 1 1122
Decoder 0 10 b token 2 B output
Decoder 1 10 b token 3 B output
Decoder 2 9 b token 1 B output
2
5
Start count, Index, Index valid, Data valid 1124
(2,i0,1,0)
(5,i1,1,0)
(6,d2,0,1)
Stage 2 1126
Stage 2 Output 0
Stage 2 Output 1
Stage 2 Output 2
Stage 2 Output 3
Preliminary Selects 1128
i0
i0 - 1
i1 - 2
i1 - 3
Stage 3 1130
Stage 3 Output 0
Stage 3 Output 1
Stage 3 Output 2
Stage 3 Output 3
Final Selects 1132
i0
i0 - 1
i1 - 2
i1 - 3
Stage 4 1134
Stage 4 Output 0
Stage 4 Output 1
Stage 4 Output 2
Stage 4 Output 3
Output Bytes 1136
Byte 0
Byte 1
Byte 2
Byte 3

Figure 42b - example using 4 input bytes, 3 decoders, 4 output bytes

09491343.012500

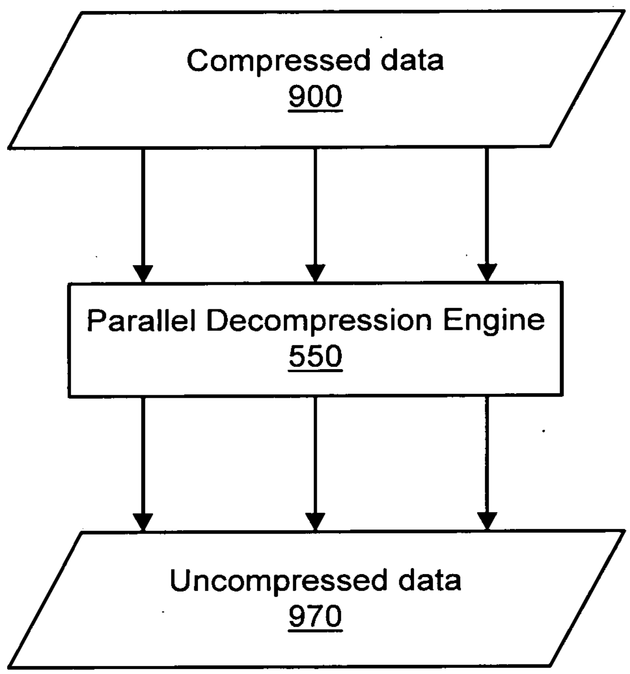


Figure 43a
Decompresion
flowchart

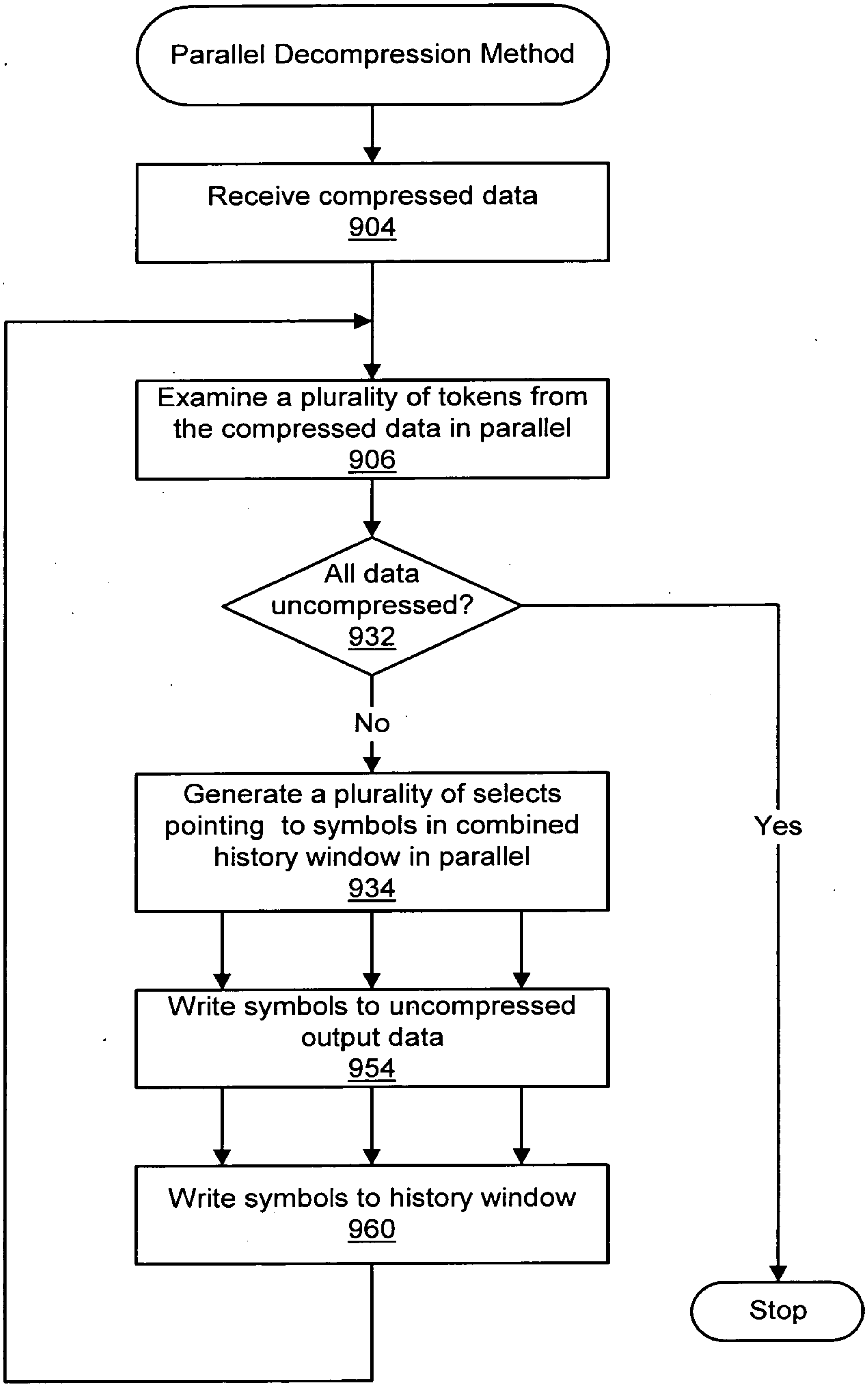


Figure 43b
Decompresion
flowchart

09471341.012500

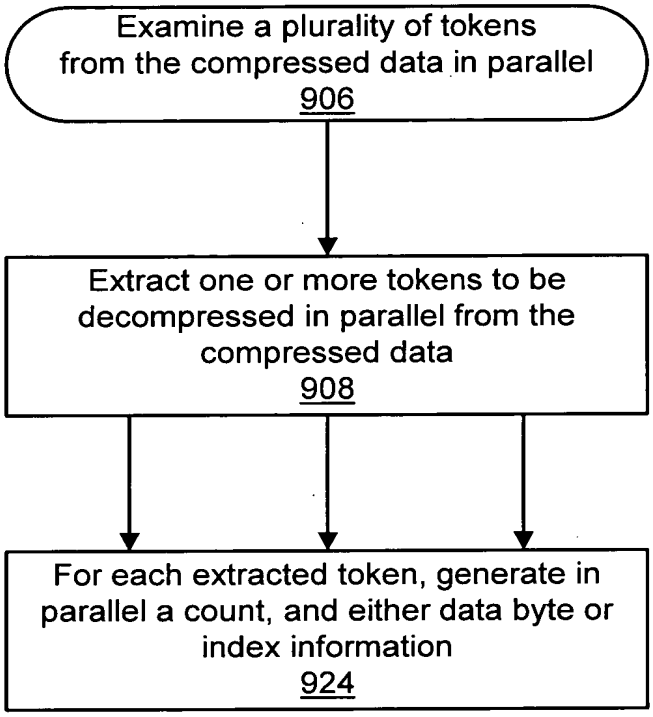


Figure 43c
Decompresion
flowchart

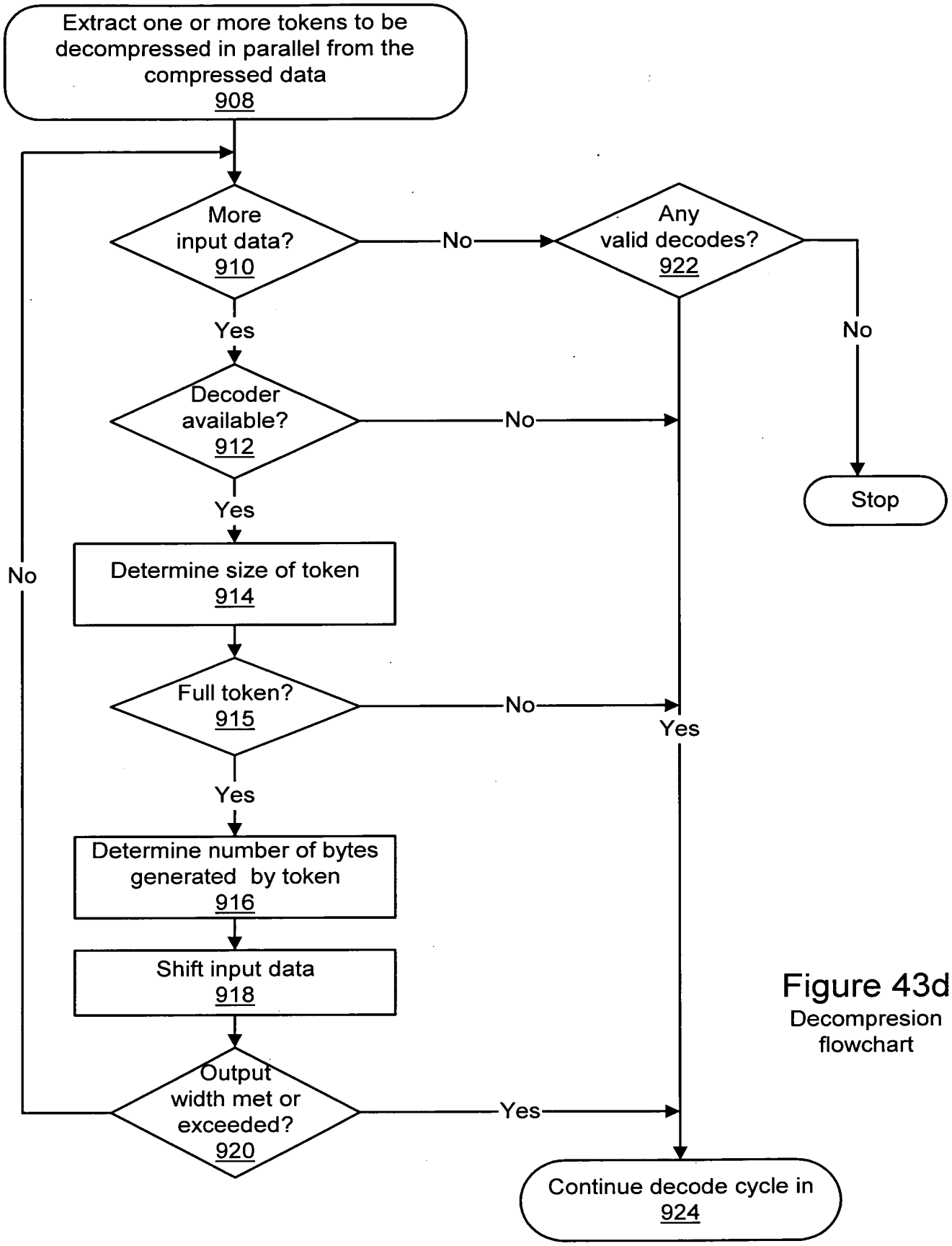


Figure 43d
Decompresion flowchart

Confidential - Software

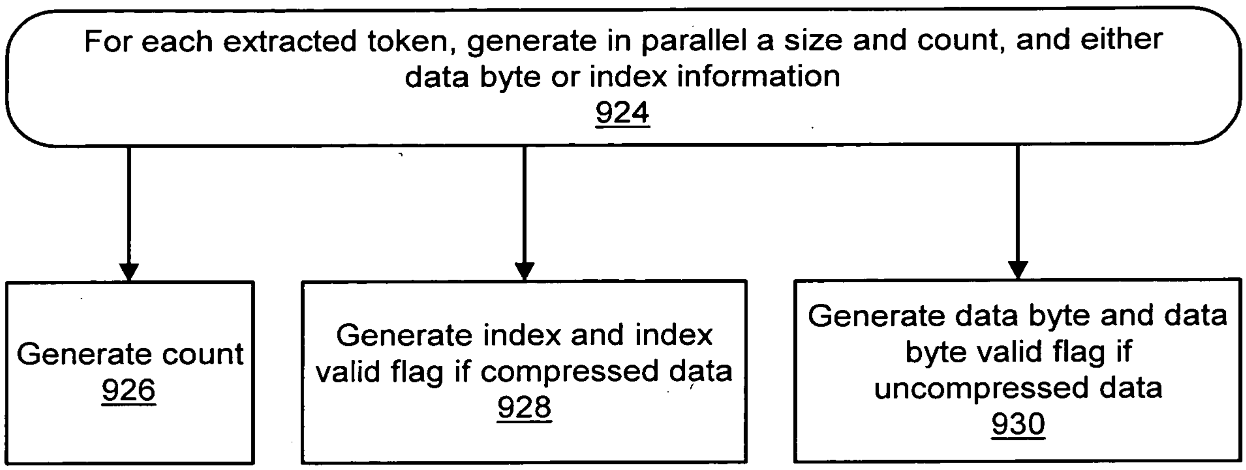


Figure 43e
Decompresion flowchart

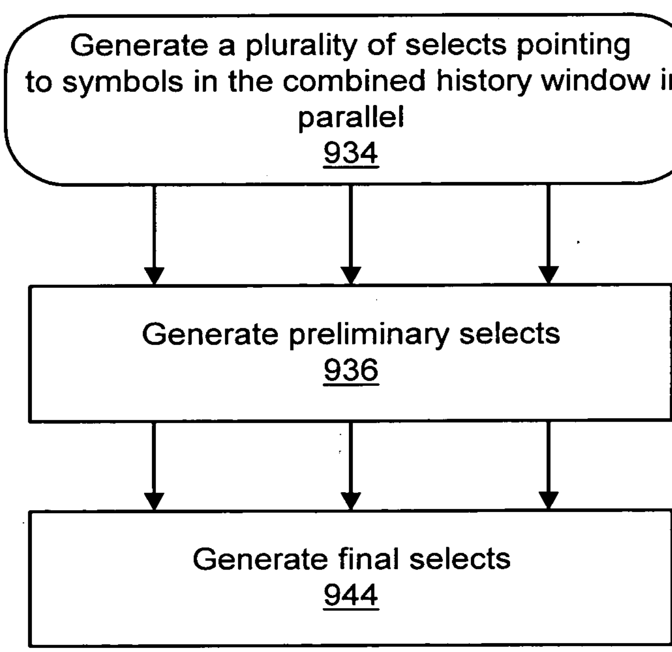


Figure 43f
Decompresion flowchart

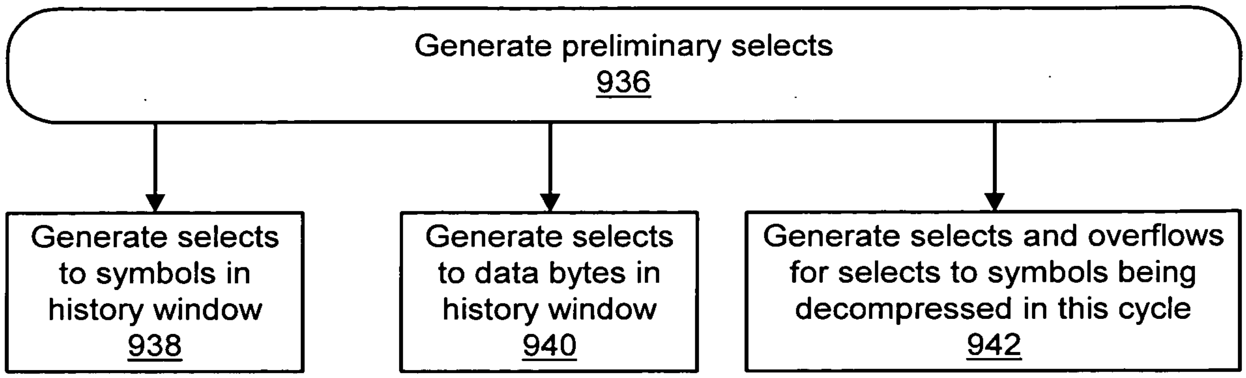


Figure 43g

Decompresion flowchart

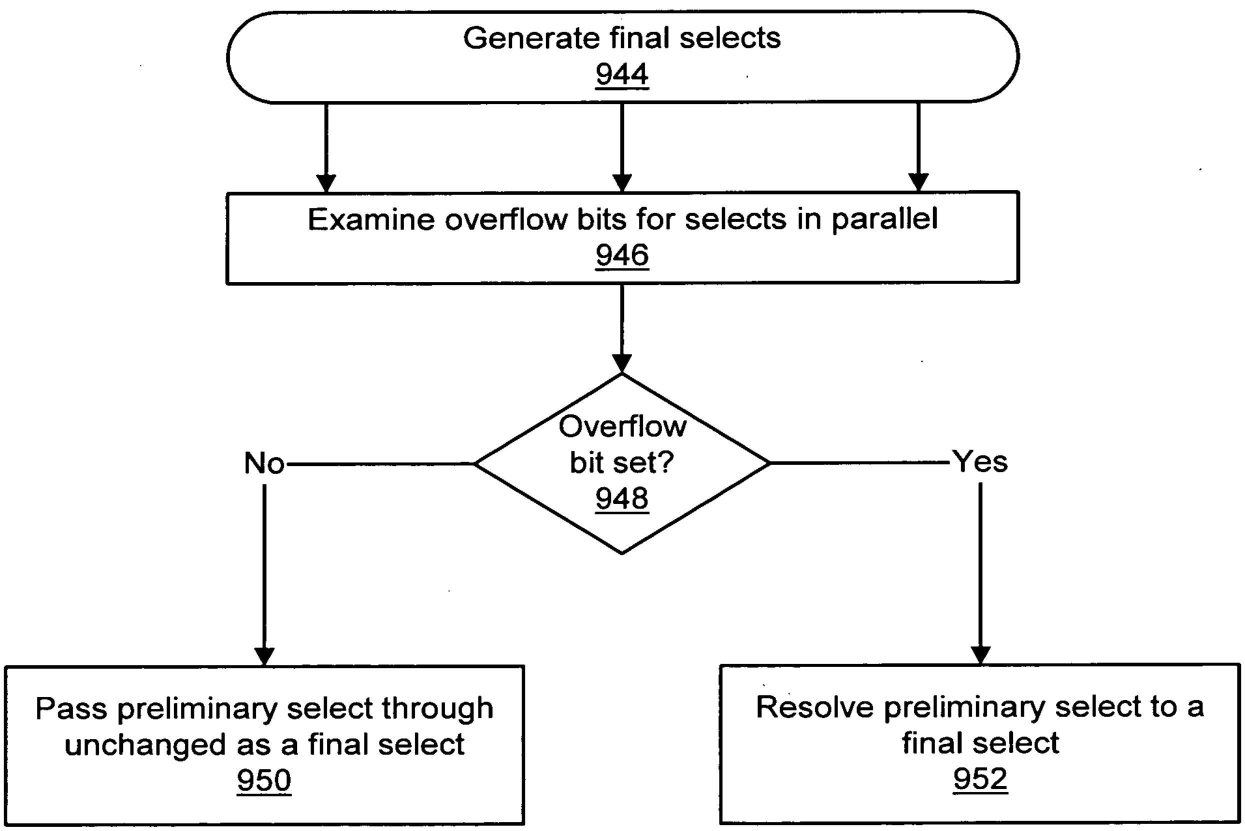


Figure 43h

Decompresion flowchart

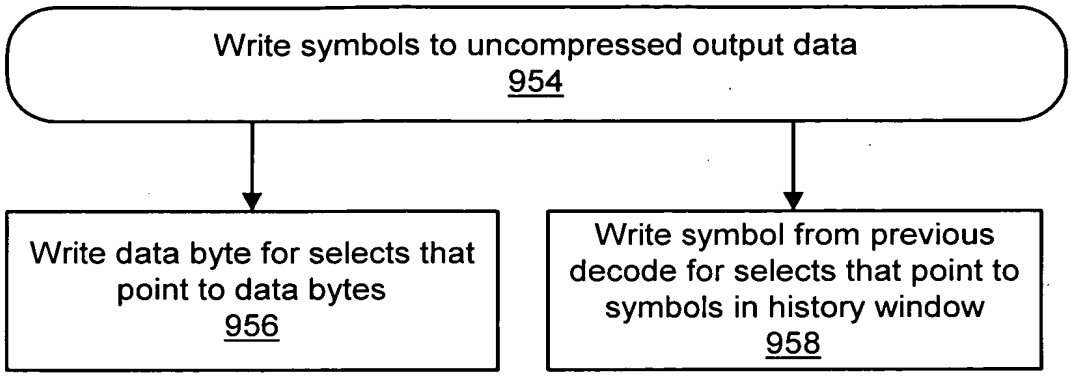


Figure 43i
Decompresion flowchart

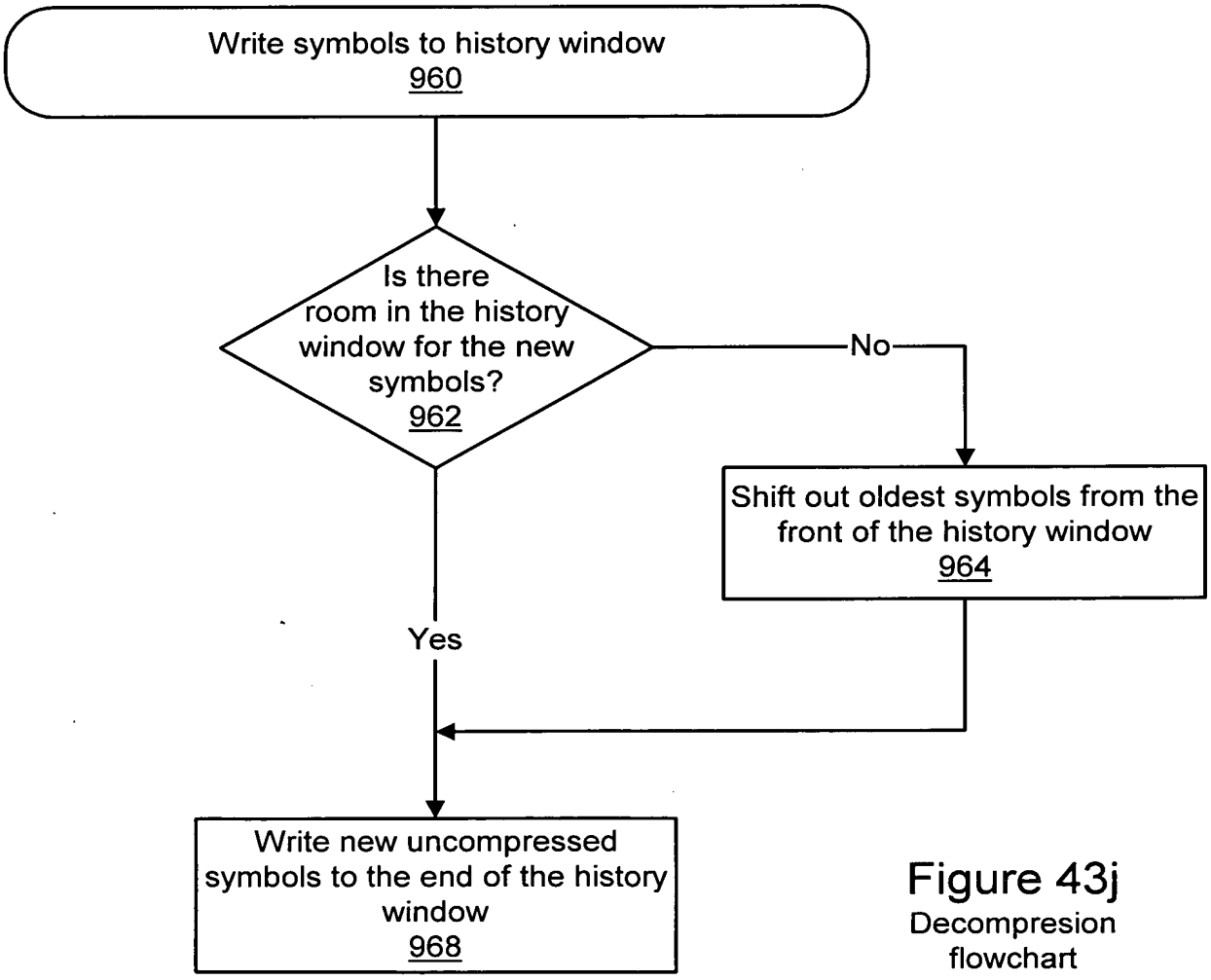
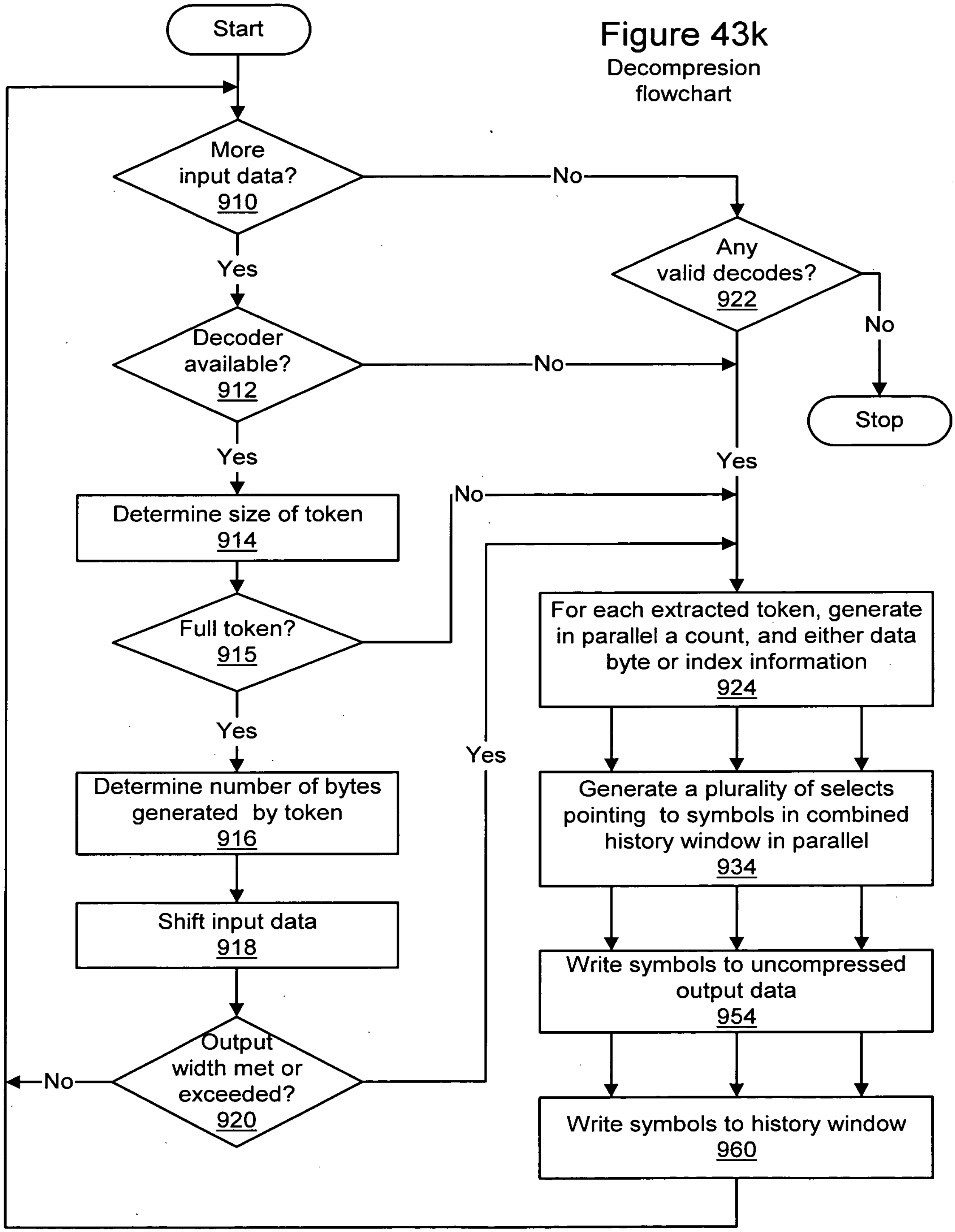


Figure 43j
Decompresion flowchart

Figure 43k
Decompresion
flowchart



Start
More input data? 910
No
Yes
Any valid decodes? 922
No
Stop
Decoder available? 912
No
Yes
Yes
Determine size of token 914
No
Full token? 915
Yes
Yes
For each extracted token, generate in parallel a count, and either data byte or index information 924
Determine number of bytes generated by token 916
Generate a plurality of selects pointing to symbols in combined history window in parallel 934
Shift input data 918
Write symbols to uncompressed output data 954
No
Output width met or exceeded? 920
Write symbols to history window 960

09491343 .012600